УДК 004.056

Имамвердиев Я.Н.
Институт Информационных Технологий НАНА, Баку, Азербайджан

yadigar@lan.ab.az

МОДЕЛЬ GM (1, 1)-МАРКОВА ДЛЯ ПРОГНОЗИРОВАНИЯ УЯЗВИМОСТЕЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Прогнозирование количества уязвимостей программного обеспечения важно для оценки рисков информационной безопасности и планирования ресурсов для быстрого устранения уязвимостей. В работе предложена модель GM (1, 1)-Маркова для прогнозирования количества уязвимостей программного обеспечения. Предложенная модель тестирована для операционной системы Microsoft XP с использованием общедоступной базы данных по уязвимостям NVD (National Vulnerability Database).

Ключевые слова: *информационная безопасность, уязвимость, прогнозирование, модель GM (1, 1)-Маркова.*

Введение

Атаки на информационные системы реализуются злоумышленником на основе той или иной уязвимости, которая присутствует в атакуемой системе. В широком смысле под уязвимостью понимается недостаток (анг. flaw) в информационной системе, используя который, можно нарушить политику информационной безопасности (ИБ). В информационных системах присутствуют различные виды уязвимостей, но уязвимости в программном обеспечении (ПО) являются одной из основных причин нарушения информационной безопасности [1].

Уязвимость ПО – один из видов недостатков ПО, предоставляющий злоумышленникам возможность обойти ограничения безопасности в целевой системе, получить несанкционированный доступ к конфиденциальной информации, намеренно нарушить ее целостность и вызвать неправильную работу системы [2]. Некоторые распространенные уязвимости и способы их активизации рассмотрены, например, в работах [2, 3]. С ростом индустрии программного обеспечения и Интернета число атак, использующих уязвимости ПО, и легкость реализации этих атак увеличиваются. Нет никакой гарантии выпустить сложный программный продукт совершенно без уязвимостей, и уязвимости неизбежно будут обнаружены после выпуска ПО. При этом крайне важно улучшить меры прогнозирования и предупреждения уязвимостей.

Большинство исследований посвящено обнаружению и предотвращению уязвимостей, однако сравнительно мало работ выполнено в области прогнозирования уязвимостей ПО. Поиск и устранение уязвимостей в ПО требуют больших трудозатрат, и адекватное прогнозирование процесса обнаружения уязвимостей позволяет разработчикам лучше планировать распределение ресурсов, необходимых для разработки и выпуска средств (патчей) уязвимостей. Быстрый процесс разработки патча позволит снизить эффект эксплойтов нулевого дня (0-дня), которые используют временное окно между обнаружением уязвимости и выпуском патча для его исправления. Он также дает метрику надежности ПО.

В этой работе разрабатывается модель прогнозирования уязвимостей безопасности программного обеспечения. Предлагается количественная модель, которая описывает скорость открытия уязвимостей в программном обеспечении. Обнаружение уязвимостей является важным компонентом идентификации рисков ИБ, и предлагаемая модель также

может быть использована для оценки рисков ИБ, связанных с открытием новых уязвимостей.

Жизненный цикл уязвимости программного обеспечения

Объективные причины появления уязвимостей в ПО заключаются в чрезвычайно высокой сложности программного кода, динамичности развития версий и легкости модификации кода [4]. К этому можно добавить проблему достоверной идентификации преднамеренно созданных программных закладок.

Большинство современных классификаций уязвимостей ПО допускают разделение их по этапам жизненного цикла ПО: проектирование (архитектура), реализация (программирование) и эксплуатация (администрирование) [4].

К уязвимостям *первого класса* относится проектирование системы без учета требований безопасности. В качестве примера можно привести сервис telnet, где имя пользователя и пароль передаются по сети в открытом виде.

К уязвимостям *второго класса* относятся, например, ошибки программирования, приводящие к переполнению буфера.

К уязвимостям *третьего класса* относятся неверное конфигурирование операционных систем, протоколов и служб, слабые пароли пользователей и др.

Каждая уязвимость уникальна, но уязвимости подчиняются определенным этапам жизненного цикла. Существуют различные модели жизненного цикла уязвимости ПО [5, 6]. Например, модель жизненного цикла, предложенная Arbaugh и соавторами, состоит из следующих семи шагов [6]:

- *Рождение:* ошибка происходит при формировании требований к ПО или при разработке ПО.
- *Обнаружение:* кто-то (хакеры, исследователи, тестировщики ПО) обнаруживает ошибку безопасности в программном обеспечении, и тогда эта ошибка становится уязвимостью. Для обнаружения уязвимостей используются методы экспертного анализа программного кода, статического и динамического анализов ПО, тесты проникновения и т.д. [7].
- *Раскрытие:* уязвимость раскрывается, когда открывший ее показывает детали проблемы. Существуют три основные политики раскрытия информации об уязвимостях [5, 8]: нераскрытие, частичное (координированное) раскрытие и полное раскрытие. Эти политики отражают точку зрения различных участников процесса раскрытия уязвимостей, и продолжаются дебаты об этических, экономических и других вопросах этого процесса [9].
- *Исправление*: уязвимость можно исправить путем разработки и выпуска исправления, рекомендаций и т.д.
- *Гласность*: уязвимость и ее проблема становятся известными.
- *Скриптинг*: выпускается эксплойт уязвимости. На этом этапе крекеры с небольшим навыком или без навыков могут использовать уязвимость для нарушения безопасности системы.
- *Смерть*: уязвимость умирает, когда применяют патч безопасности для всех уязвимых систем.

Рисунок 1 иллюстрирует переход состояний типичной уязвимости в модели жизненного цикла.

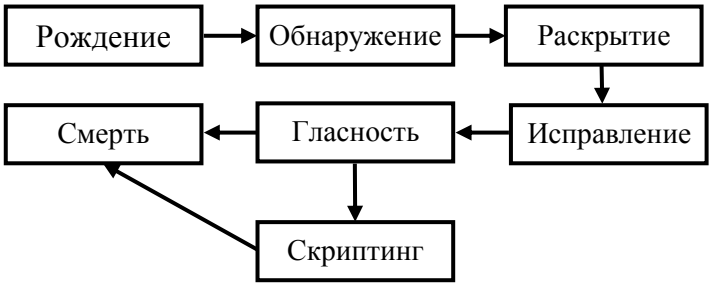

Рис. 1. Типичные переходы состояний в жизненном цикле

Отметим, что так называемые уязвимости 0-дня в ПО – это уязвимости, которые эксплуатируются злоумышленниками задолго до того, как производитель ПО узнает о том, что они присутствуют в его продуктах [10].

Как правило, информация о таких уязвимостях хранится в строжайшей тайне и является крайне ценной для злоумышленников [11, 12]. Однако как только эти данные становятся общедоступными, большинство разработчиков публикует обновления безопасности или инструкции по предотвращению атак с использованием обнаруженных брешей.

Знание новых уязвимостей дает киберпреступникам возможность для атаки на любую цель по своему выбору, оставаясь при этом незамеченными. К сожалению, эти серьезные угрозы трудно анализировать, потому что, в общем, данные не доступны до обнаружения атак 0-дня. Кроме того, атаки 0-дня являются редкими событиями, которые вряд ли можно наблюдать в «приманке» (honeypot) или в лабораторных экспериментах [13].

Обзор существующих работ

В работе [14] предложена так называемая термодинамическая модель на основе моделей роста надежности ПО (Software Reliability Growth Models), в которых предполагается, что в процессе тестирования надежность ПО увеличится. В термодинамической модели вероятность сбоя безопасности в момент времени t, когда были удалены n дефектов, обратно пропорциональна t для альфа-тестеров. Эта вероятность еще ниже для бета-тестеров (они не имеют доступа к исходным кодам ПО). Отметим, что автор не проводил экспериментов для проверки предложенной модели.

В работе [15] предложены две модели: линейная модель и экспоненциальная модель. Автор проводил эксперименты на четырех версиях разных операционных систем (Windows NT 4.0, Solaris 2.5.1, Free BSD 4.0 and RedHat 7.0). Во всех этих случаях модели хорошо согласовывались с данными (p-значение изменялось в интервале от 0.167 до 0.589).

В работе [16] Alhazmi и Malaiya предложили так называемую AML-модель (Alhazmi&Malaias Logistic Model), исходя из S-образной модели роста надежности ПО. Основной идеей авторов является деление процесса обнаружения уязвимостей на три фазы: фаза изучения, линейная фаза и фаза насыщения. В первой фазе требуется некоторое время, чтобы люди изучали программное обеспечение, поэтому обнаруживается меньше уязвимостей. На втором этапе, когда люди получают более глубокие знания о программном обеспечении, обнаруживается гораздо больше уязвимостей. На заключительном этапе программное обеспечение устаревает и не так много людей будут его использовать. Люди теряют интерес к нахождению новых уязвимостей. Таким образом кумулятивное число уязвимостей является стабильным.

Существует несколько работ, посвященных эмпирическому анализу предложенных моделей обнаружения уязвимостей [16–21]. AML-модель была предметом значительной экспериментальной проверки: для операционных систем [19], браузеров (IE, Firefox, Mozilla) [16] и веб-серверов (ISS, Apache) [21]. Результаты, представленные в литературе,

показывают, что нет достаточных доказательств, чтобы принимать или отвергать AML [22, 23].

Ozment в работе [24] рассмотрел модели обнаружения уязвимостей (предложенные Alhazmi-Malaiya [16, 17]) и указал на некоторые ограничения, которые делают эти модели неприменимыми. Одним из ограничений является нехватка информации, включенной в базу данных уязвимостей (например, National Vulnerability Database).

Несколько работ посвящено эмпирическому исследованию использования баз данных уязвимостей для прогнозирования уязвимостей ПО [25–27]. Massacci и соавторы [27] сравнили несколько существующих баз данных уязвимостей на основе признаков уязвимостей, доступных в каждой базе. Они указали, что многие важные признаки, например время обнаружения, не включены в большинство баз данных. Хотя провайдеры некоторых баз данных утверждают, что все признаки включены, однако часто часть полей оказывается пустой. Massacci и соавторы [27] показали, что при использовании двух различных источников данных в одинаковых экспериментах результаты могут быть совершенно разными в связи с высокой степенью несогласованности данных, доступных научному сообществу в настоящее время.

McQueen и соавторы [28] предложили алгоритмы для оценки числа уязвимостей 0-дня для каждого заданного дня. Это число может характеризовать общий уровень риска от уязвимостей нулевого дня. Однако для разных приложений эти риски могут быть различными.

Исследователи из Symantec Research Labs разработали метод автоматической идентификации атак, использующих уязвимости 0-дня [29]. Базой для экспериментов послужили данные, собранные в период с 2008 по 2011 год из 11 миллионов реальных хостов по всему миру, использующих антивирусное программное обеспечение компании. В экспериментах были обнаружены 18 уязвимостей 0-дня, 11 из которых не использовались ранее в целенаправленных атаках. Исследователи обнаружили, что типичная атака 0-дня в среднем длится около 312 дней, а после публичного раскрытия информации об уязвимости число атак, эксплуатирующих ее, увеличивается примерно в пять раз. Эксперты также подчеркивают, что на самом деле уязвимости 0-дня являются более распространенными и опасными, чем принято считать.

Модель GM (1, 1)

Модель GM (1, 1) часто используется в различных приложениях, в частности для прогнозирования [30–34]. По сравнению со статистическими методами, модель GM (1, 1) имеет некоторые преимущества. Например, доказано, что модель GM (1, 1) становится эффективной при размере временного ряда больше 4. Также необязательно строить предложения о статистическом распределении данных [30, 31].

Данные должны располагаться через равный временной интервал в последовательном порядке без пропусков. Процесс построения модели GM (1, 1) приводится ниже [31].

Обозначим временной ряд данных как

$$x^{(0)} = \left(x^{(0)}(1), x^{(0)}(2),..., x^{(0)}(n)\right), \quad (1)$$

где n – количество данных в последовательности.

Из этих данных генерируется возрастающая последовательность $x^{(1)}$ с помощью операции AGO (Accumulated Generating Operating) следующим образом:

$$x^{(1)} = \left(x^{(1)}(1), x^{(1)}(2),..., x^{(1)}(n)\right), \quad (2)$$

где $x^{(1)}(k) = \sum_{j=1}^{k} x^{(0)}(j), \quad k = 1,2,...,n.$

Основная цель операции AGO – уменьшить случайность данных.

Модель GM (1, 1) строится с помощью дифференциального уравнения первого порядка для $x^{(1)}(k)$:

$$\frac{dx^{(1)}(k)}{dk} + ax^{(1)}(k) = b, \tag{3}$$

где параметры *a* и *b* называют коэффициентами развития и серым входом соответственно.

Решение уравнения (3) находится с помощью метода наименьших квадратов:

$$\hat{x}^{(1)}(k) = \left(x^{(0)}(1) - \frac{\hat{b}}{\hat{a}} \right) e^{-\hat{a}(k-1)} + \frac{\hat{b}}{\hat{a}}, \tag{4}$$

где $\left[\hat{a}, \hat{b}\right]^T = \left(B^T B\right)^{-1} B^T Y$ и

$$B = \begin{bmatrix} -0.5\left(x^{(1)}(1) + x^{(1)}(2)\right) & 1 \\ -0.5\left(x^{(1)}(2) + x^{(1)}(3)\right) & 1 \\ \cdots & \cdots \\ -0.5\left(x^{(1)}(n-1) + x^{(1)}(n)\right) & 1 \end{bmatrix}$$

$$Y = \left[x^{(0)}(2), x^{(0)}(3), ..., x^{(0)}(n)\right]^T.$$

Применив обратный оператор AGO, получим:

$$\hat{x}^{(0)}(k) = \left(x^{(0)}(1) - \frac{\hat{b}}{a} \right)\left(1 - e^{\hat{a}}\right) e^{-\hat{a}(k-1)},\ k = 2, 3, ..., \tag{5}$$

где $\hat{x}^{(0)}(1) = x^0(1)$.

$\hat{x}^{(0)}(1), \hat{x}^{(0)}(2), ..., \hat{x}^{(0)}(n)$ называют последовательностью, удовлетворяющей модели GM (1, 1), а $\hat{x}^{(0)}(n+1), \hat{x}^{(0)}(n+2), ...,$ прогнозными значениями GM (1, 1).

Модель GM (1, 1)-Маркова

Идея модели GM (1, 1)-Маркова в том, чтобы создать механизм переходов между остаточными значениями (ошибками) с помощью матриц вероятностей переходов и затем применять возможную поправку при прогнозировании [35, 36].

Вектор остаточных значений (ошибок) $E = \{e_1, e_2, ..., e_n\}$ определим формулой

$$e_i = x^0(i) - \hat{x}^{(0)}(i),\ i = 1, 2, ..., n. \tag{6}$$

Для создания матриц переходов между состояниями Маркова и прогнозирования остаточных значений используется нижеприведенный процесс [37]:

1*) Определение состояний:* остаточные значения (6) разбиваются на *s* состояний R_1, R_2, …, R_s таким образом, чтобы каждое состояние имело примерно равное число данных.

2) *Построение матрицы вероятностей перехода:* t_1, t_2, …, t_s называются временами перехода, и P_{ij} определяется как вероятность перехода из состояния *i* в состояние *j* за один шаг. Матрица *P* определяется как матрица вероятностей перехода:

$$\mathrm{P} = \begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1s} \\ P_{21} & P_{22} & \cdots & P_{2s} \\ \vdots & \vdots & \vdots & \vdots \\ P_{s1} & P_{s2} & \cdots & P_{ss} \end{bmatrix}. \tag{7}$$

Как известно, любое состояние R_i будет переходить в одно из состояний R_1, R_2, …, R_s. Поэтому $\sum_j P_{ij} = 1 \, (0 \le P_{ij} \le 1, \, i = 1,2,...)$.

Если состояние R_i переходит в состояние R_j за два шага, три шага, … или m шагов, то соответствующие вероятности перехода называются вероятностями перехода второго порядка, …, m-го порядка ($P_{ij}^{(2)}, P_{ij}^{(3)},..., P_{ij}^{(m)}$).

Вероятность перехода из состояния i в состояние j после m шагов вычисляется как:

$$P_{ij}^{(m)} = \frac{M_{ij}^{(m)}}{M_i}, \qquad i, j = 1,2,..., s, \tag{8}$$

где $M_{ij}^{(m)}$ – число переходов из состояния i в состояние j после m шагов и M_i – число всех состояний, которые являются состоянием i.

Матрица переходов между s состояниями для m-шагов задается:

$$R^{(m)} = \begin{bmatrix} P_{11}^{(m)} & P_{12}^{(m)} & ... & P_{1s}^{(m)} \\ P_{21}^{(m)} & P_{22}^{(m)} & ... & P_{2s}^{(m)} \\ \cdots & \cdots & \cdots & \cdots \\ P_{s1}^{(m)} & P_{s2}^{(m)} & ... & P_{ss}^{(m)} \end{bmatrix}. \tag{9}$$

3) *Вычисление сумм вероятностей перехода:* выбирая s ближайших переходов от времени прогнозирования, определяются шаги перехода как один шаг, два шага ... и s шагов до времени прогнозирования соответственно. В соответствующих матрицах перехода соответствующие векторы – строки начальных состояний дают вероятность того, что появляется каждое состояние, и таким образом можно вычислить сумму каждой вероятности перехода.

4) *Выбор состояния:* состояние, которое имеет наибольшую сумму вероятностей перехода, соответствует состоянию остаточного значения. Интервалом изменения прогнозного значения будет выбранное состояние $R_i = [R_{i-}, R_{i+}]$.

5) *Вычисление остаточного значения для прогнозирования:* остаточное значение можно определить как медиану в $[R_{i-}, R_{i+}]$:

$$\hat{e}_i = 0.5(R_{i-} + R_{i+}). \tag{10}$$

6) *Прогнозирование:* прогнозное значение $\hat{y}(i)$ вычисляется следующим образом:

$$\hat{y}(i) = \hat{x}^{(0)}(i) + \hat{e}_i. \tag{11}$$

Базы данных уязвимостей для вычислительных экспериментов

Существует несколько открытых баз данных уязвимостей безопасности программного обеспечения [38–41]: NVD (National Vulnerability Database), OSVDB (Open Source Vulnerability Database), ZDI (Zero Day Initiative), базы данных уязвимостей Symantec и т.д. В вычислительных экспериментах по тестированию предложенной модели использовалась база данных уязвимостей NVD [38], которая поддерживается Национальным институтом стандартов и технологий США с 2005 г. Репозитарий NVD содержит базы данных уязвимостей безопасности ПО, проверочных листов безопасности, ошибок конфигурации, названий продуктов и оценки уязвимостей. NVD объединяет всю доступную правительству США информацию в единой базе, основывается на стандарте CVE и полностью соответствует ему. CVE (Common Vulnerabilities and Exposures) – тезаурус и база данных уязвимостей ПО – определяют единые правила именования уязвимостей и открыты всем заинтересованным лицам (на сайте корпорации MITRE).

На веб-сайте NVD доступны сервисы поиска, статистики и загрузки данных об уязвимостях в формате XML. На 19 февраля 2014 года в репозитарии NVD имелись 60 597 CVE уязвимостей.

В базе данных NVD каждая уязвимость оценивается по метрике CVSS (Common Vulnerability Scoring System – общая система оценивания уязвимостей) [41]. Оценка CVSS представляет собой числовую величину от 0 до 10, где 10 – максимальный уровень опасности, соответствующей критической уязвимости.

В таблице 1 и на рисунке 2 приведены данные о числах обнаруженных уязвимостей Microsoft XP по базе данных NVD.

Таблица 1

Число обнаруженных уязвимостей Microsoft XP

Годы	Число уязвимостей	Годы	Число уязвимостей
2001	10	2008	75
2002	34	2009	128
2003	23	2010	237
2004	44	2011	246
2005	67	2012	109
2006	62	2013	152
2007	116		

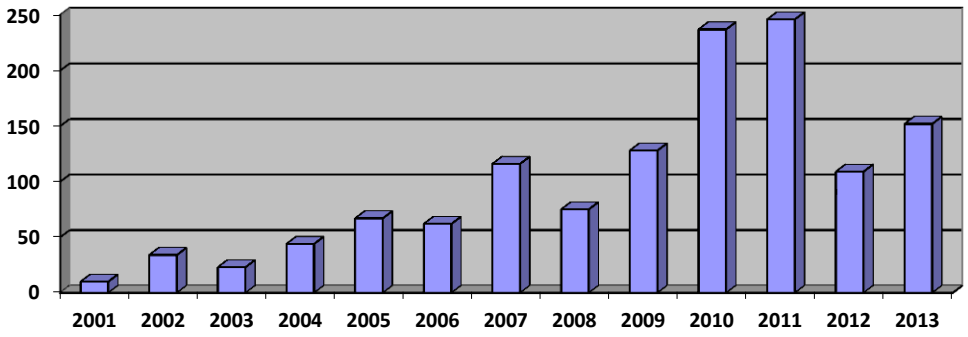

Рис. 2. Число обнаруженных уязвимостей Microsoft XP по годам

Результаты вычислительных экспериментов

В этом разделе сравниваются модели GM (1, 1) и GM(1, 1)-Маркова. Данные из базы NVD о числе обнаруженных уязвимостей в программном обеспечении Microsoft XP с 2002 по 2010 год были использованы для оценки эффективности и практичности этих моделей. Данные с 2004 по 2009 год были использованы для построения моделей GM (1, 1) и GM(1, 1)-Маркова, а данные с 2010 по 2013 год для проверки точности модели.

Реальные и прогнозируемые числа уязвимостей приводятся в таблице 2.

По предложенной схеме построим матрицы переходов между состояниями для модели GM(1, 1)-Маркова. Будем рассматривать s=4 состояний для каждого шага по времени. Чтобы определить состояния, разобьем интервал между наименьшим и наибольшим остаточными значениями на четыре диапазона: R_1 = [-78; -28], R_2 = [-28; 3], R_3 = [10; 99].

Таблица 2

Реальные и прогнозируемые по GM(1,1) числа уязвимостей Microsoft XP

	Годы	Номер года	Число уязвимостей		
			Реальные	GM(1,1)	Остаточные значения
Построение модели	2004	1	44	44	0
	2005	2	67	64	3
	2006	3	62	74	-12
	2007	4	116	87	-29
	2008	5	75	101	-26
	2009	6	128	118	10
	2010	7	237	138	99
Прогнозирование по модели	2011	8	246	161	85
	2012	9	109	187	-78
	2013	10	152	218	-66

В таблице 3 приведены номера годов и соответствующие состояния.

Таблица 3

Номера годов и соответствующие состояния

Номер года	1	2	3	4	5	6	7	8	9	10
Состояние	R_2	R_2	R_2	R_1	R_2	R_3	R_3	R_3	R_1	R_1

Используя таблицу 3, по формулам (8)–(9) можно вычислить следующие матрицы переходов размерности 3x3:

$$P^{(1)} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/4 & 1/2 & 1/4 \\ 1/3 & 0 & 2/3 \end{bmatrix}, \; P^{(2)} = \begin{bmatrix} 0 & 0 & 1 \\ 1/4 & 1/2 & 1/4 \\ 1/2 & 0 & 1/2 \end{bmatrix}, P^{(3)} = \begin{bmatrix} 0 & 0 & 1 \\ 1/4 & 1/4 & 1/2 \\ 1 & 0 & 0 \end{bmatrix}.$$

Далее, для примера прогнозируем число уязвимостей Microsoft XP, которые могут быть обнаруженными в 2010 году. Для этого возьмем данные по трем ближайшим годам и вычислим суммы вероятностей перехода, результаты приведены в таблице 4.

Таблица 4

Вычисление суммы вероятностей перехода

Состояние перехода R_i		R_1	R_2	R_3
Номер года	Начальное состояние			
4	R_1	0	0	1
5	R_2	1/4	1/2	1/4
6	R_3	1/3	0	2/3
Сумма вероятностей перехода		7/12	1/2	11/12

Из таблицы 4 видно, что сумма вероятностей перехода из любого состояния в состояние R_3 максимальна (11/12). Поэтому диапазон остаточного значения в 2010 году является $R_3 = [10, 99]$. Медиан этого интервала равен 44.5, по формуле (11) находим прогнозное значение для этого года 183 (=138+45). Таблица 5 представляет значения, вычисленные по модели.

Таблица 5

Реальные и прогнозируемые по модели GM(1, 1)-Маркова числа уязвимостей Microsoft XP

Годы	Номер года	Число уязвимостей			
		Реальные	GM(1, 1)	GM(1, 1)-Марков	Остаточные значения
2004	1	44	44	44	0
2005	2	67	64	51	16
2006	3	62	74	61	1
2007	4	116	87	74	42
2008	5	75	101	88	-13
2009	6	128	118	163	-35
2010	7	237	138	183	54
2011	8	246	161	206	40
2012	9	109	187	162	-53
2013	10	152	218	193	-41

Для оценки точности предложенных моделей прогнозирования в этой работе используются два показателя оценки: средняя абсолютная ошибка в процентах (Mean Absolute Percentage Error) и среднеквадратическая ошибка (Root Mean Square Error). Эти показатели вычисляются по формулам (12) и (13) соответственно.

1. MAPE (Mean Absolute Percentage Error)

$$MAPE = \frac{1}{n}\sum_{k=1}^{n}\frac{\left|x^{(0)}(k) - \hat{x}^{(0)}(k)\right|}{x^{(0)}(k)}. \quad (12)$$

2. RMSE (Root Mean Square Error)

$$RMSE = \sqrt{\frac{\sum_{k=1}^{n}\left(x^{(0)}(k) - \hat{x}^{(0)}(k)\right)^2}{n}}, \quad (13)$$

где $\hat{x}^{(0)}(k)$ – ожидаемое значение для периода k, $x^{(0)}(k)$ – фактическое значение для периода k, n – число периодов.

Чем меньше значения вышеприведенных показателей, тем лучше модели прогнозирования; меньшие значения означают, что вычисленные результаты ближе к реальным данным.

Соответствующие усредненные значения ошибок прогнозирования предложенных моделей приведены в таблице 6.

Таблица 6

Оценки точности предложенных моделей прогнозирования

Этапы	GM(1,1)		GM(1,1)-Маркова	
	MAPE	RMSE	MAPE	RMSE
2004-2009	15.21	17.18	17.73	23.86
2010-2013	47.83	82.86	28.66	47.45

Ошибки прогнозирования модели GM(1,1)-Маркова сравнительно малы и ее можно использовать для краткосрочных прогнозов, поэтому модель GM(1,1)-Маркова была использована для прогнозирования числа уязвимостей программного обеспечения Microsoft XP в 2014 и 2015 годах. Результаты прогноза по модели приведены в таблице 7.

Таблица 7

Прогнозируемые числа уязвимостей Microsoft XP в 2014 и 2015 годах

Годы	2010	2011	2012	2013	2014	2015
Реальные значения	237	246	109	152		
Прогнозные значения	183	206	162	193	230	272

Заключение

Уязвимости программного обеспечения являются основными факторами рисков информационной безопасности. Для предотвращения и предупреждения этих рисков необходимы более точные модели прогнозирования уязвимостей. В этой работе предложена модель прогнозирования числа уязвимостей, которые могут быть обнаружены в определенный период времени в данном программном обеспечении. Возможности прогнозирования проверены для операционной системы Microsoft XP на основе информации из общедоступной базы данных об уязвимостях. Прогнозирование количества уязвимостей позволяет разработчикам адекватно планировать распределение ресурсов, требуемых для устранения уязвимостей. Оно также важно для планирования ресурсов команд быстрого реагирования (команды CERT) на обработку инцидентов информационной безопасности. Поставщики и пользователи программного обеспечения могут использовать предлагаемый подход в качестве руководящих принципов для прогнозирования уровня критичности и частоты будущих уязвимостей.

Литература

1. Igure V., Williams R. Taxonomies of attacks and vulnerabilities in computer systems // IEEE Communications Surveys & Tutorials, 2008, vol.10, no.1, pp.6–19.
2. Hoglund G., McGraw G. Exploiting Software: How to Break Code. Addison-Wesley Professional, 2004.
3. Whittaker J.A., Thompson H.H. How to Break Software Security: Effective Techniques for Security Testing. Pearson, 2003
4. Марков А.С., Фадин А.А. Систематика уязвимостей и дефектов безопасности программных ресурсов // Защита информации. INSIDE, 2013, №3, с.2–7.
5. National Infrastructure Advisory Council: Vulnerability Disclosure Framework, 2004.
6. Arbaugh W.A., Fithen W.L., McHugh J. Windows of vulnerability: A case study analysis // IEEE Computer, 2000, vol.33, no.12, pp.52–59.
7. Sezer E.C., Kil Ch., Ning P. Automated software vulnerability analysis. Advances in Information Security, 2010, vol.46, pp.201–223.

8. Takanen A., Vuorijärvi P., Laakso M., Röning J. Agents of responsibility in software vulnerability processes // Ethics and Information Technology, 2004, vol.6, no.2, pp.93–110.
9. Cavusoglu H., Cavusoglu H., Raghunathan S. Efficiency of vulnerability disclosure mechanisms to disseminate vulnerability knowledge // IEEE Transactions on Software Engineering, 2007, vol.33, no.3, pp.171–185.
10. Everett C. Zero-day, but not zero-risk // Infosecurity, 2007, vol.4, no.7, pp.36–39.
11. Miller C. The legitimate vulnerability market: Inside the secretive world of 0-day exploit sales / Proc. of the 6th Workshop on the Economics of Information Security, 2007, pp.1–10.
12. Radianti J., Gonzalez J.J. A preliminary model of the vulnerability black market / Proc. of the 25th International Conference of the System Dynamics Society, 2007, pp.1–30.
13. Evans N., Yuan X. Observation of recent Microsoft zero-day vulnerabilities / Proc. of the 49th Annual ACM Southeast Regional Conference, 2011, pp.328–329.
14. Anderson R. Security in open versus closed systems - the dance of Boltzmann, Coase and Moore / Proc. of Open Source Software: Economics, Law and Policy, 2002, pp.1–13.
15. Rescorla E. Is finding security holes a good idea? // IEEE Security and Privacy, 2005, vol.3, no.1, pp.14–19.
16. Alhazmi O.H., Malaiya Y.K. Modeling the vulnerability discovery process / Proc. of the 16th IEEE International Symposium on Software Reliability Engineering, 2005, pp.129–138.
17. Alhazmi O.H., Malaiya Y.K. Prediction capabilities of vulnerability discovery models / Proc. of the Annual Reliability and Maintainability Symposium, 2006, pp.86–91.
18. Alhazmi O.H., Malaiya Y.K., Ray I. Measuring, analyzing and predicting security vulnerabilities in software systems // Computers & Security, 2007, vol.26, no.3, pp.219–228.
19. Alhazmi O.H., Malaiya Y.K. Application of vulnerability discovery models to major operating systems // IEEE Transactions on Reliability, 2008, vol.57, no.1, pp.14–22.
20. Woo S.-W., Alhazmi O.H., Malaiya Y.K. An analysis of the vulnerability discovery process in web browsers / Proc. of 10th IASTED International Conference on Software Engineering and Applications, 2006, pp.172–177.
21. Woo S.-W., Joh H., Alhazmi O.H., Malaiya Y.K. Modeling vulnerability discovery process in Apache and IIS HTTP servers // Computers & Security, 2011, vol.30, no.1, pp.50–62.
22. Neuhaus S., Zimmermann T., Holler C., Zeller A. Predicting vulnerable software components / Proc. of the 14th ACM Conference on Computer and Communications Security, 2007, pp.529–540.
23. Nguyen V.H., Massacci F. An independent validation of vulnerability discovery models / Proc. of the 8th ACM Symposium on Information, Computer and Communications Security, 2012, pp.6–7.
24. Ozment A. Improving vulnerability discovery models / Proc. of the ACM Workshop on Quality of Protection, 2007, pp.6–11.
25. Okamura H., Tokuzane M., Dohi T. Quantitative security evaluation for software system from vulnerability database // Journal of Software Engineering and Applications, 2013, vol.6, no.4A, pp.15–23.
26. Zhang S., Caragea D., Ou X. An empirical study on using the National Vulnerability Database to predict software vulnerabilities / Proc. of the 22nd International Conference on Database and Expert Systems Applications, 2011, Part I, pp.217–231.
27. Nguyen V. H., Massacci F. The (un)reliability of NVD vulnerable versions data: an empirical experiment on Google Chrome vulnerabilities / Proc. of the 8th ACM Symposium on Information, Computer and Communications Security, 2013, pp.493–498.
28. McQueen M.A., McQueen T.A., Boyer W.F., Chan M.R. Empirical estimates and observations of 0day vulnerabilities / Proc. of the 42nd Hawaii International Conference on System Sciences, 2009, pp.1–12.

29. Bilge L., Dumitras T. Before we knew it: An empirical study of zero-day attacks in the real world / Proc. of the ACM conference on Computer and Communications Security, 2012, pp.833–844.
30. Deng J. Introduction to grey system theory // Journal of Grey System, 1989, vol.1, no.1, pp.1–24.
31. Liu S., Lin Y. Grey systems theory and applications. Springer-Verlag Berlin Heidelberg. 2011, 379 p.
32. Kayacan E., Kaynak O., Ulutas B. Grey system theory-based models in time series prediction // Expert Systems with Application, 2010, vol.37, no.2, pp.1784–1789.
33. Wang Y.F. Predicting stock price using fuzzy grey prediction system // Expert Systems with Applications, 2002, vol.22, no.1, pp.33–39.
34. Chiu N.H. An early software-quality classification based on improved grey relational classifier // Expert Systems with Applications, 2009, vol.36, no.7, pp.10727–10734.
35. Zhang Y. Predicting model of traffic volume based on Grey-Markov // Modern Applied Science, 2010, vol.4, no.3, pp.46–50.
36. Hsu C.C., Chen C.Y. Application of improved grey prediction model for power demand forecasting // Energy Conversion and Management, 2003, vol.44, no.14, pp.2241-2249.
37. Li C. Grey Markov model based on parameter fits and its application in stock price prediction / Proc. of the 6th International Conference on Intelligent Systems Design and Applications, 2006, vol.1, pp.594-598.
38. National Vulnerability Database (NVD). http://nvd.nist.gov/home.cfm
39. OSVDB. The open source vulnerability database. http://www.osvdb.org/, 2012.
40. TippingPoint: The Zero Day Initiative (ZDI). http://www.zerodayinitiative.com/
41. Mell P., Scarfone K., Romansky S. A Complete Guide to the Common Vulnerability Scoring System Version 2.0. Forum of Incident Response and Security Teams, June 2007. www.first.org/cvss/cvss-guide.html.

UOT 004.056
Yadigar İmamverdiyev N.
AMEA İnformasiya Texnologiyaları İnstitutu, Bakı, Azərbaycan
yadigar@lan.ab.az
Proqram təminatı boşluqlarının proqnozlaşdırılması üçün GM (1, 1)-Markov modeli
Proqram təminatı boşluqlarının sayının proqnozlaşdırılması informasiya təhlükəsizliyi risklərinin qiymətləndirilməsi və boşluqların qısa müddətdə aradan qaldırılması məqsədilə resursların planlaşdırılması üçün vacibdir. Bu işdə proqram təminatı boşluqlarını verilmiş zaman müddətində proqnozlaşdırmaq üçün GM (1, 1)-Markov modeli təklif edilir. Təklif olunan model boşluqların açıq verilənlər bazası NVD (National Vulnerability Database) istifadə edilərək Microsoft XP əməliyyat sistemi üçün test edilmişdir.
***Açar sözlər**: informasiya təhlükəsizliyi; boşluq; proqnozlaşdırma; GM(1,1)-Markov modeli.*

Yadigar N. Imamverdiyev
Institute of Information Technology of ANAS, Baku, Azerbaijan
yadigar@lan.ab.az
GM (1, 1)-Markov model for software vulnerabilities prediction
Prediction of the number of software vulnerabilities is important for assessing information security risks and resource planning for the rapid elimination of vulnerabilities. This paper proposes a GM(1, 1)-Markov model to predict the number of software vulnerabilities for given time period. The proposed model is tested for Microsoft XP operating system using a publicly available vulnerability database – the NVD (National Vulnerability Database).
***Keywords:** information security; vulnerability; prediction; GM (1, 1)-Markov model.*