УДК 004.056.57

Вестник УрФО № 3(49) / 2023, с. 42–53

Гибилинда Р. В.

DOI: 10.14529/secur230304

МОДЕЛЬ ПРОЦЕССА ВЫЯВЛЕНИЯ ИНФОРМАЦИИ О ЗАПУСКЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ В ОПЕРАЦИОННЫХ СИСТЕМАХ WINDOWS ПРИ РАССЛЕДОВАНИИ ИНЦИДЕНТОВ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

В статье представлена основанная на математическом аппарате сетей Петри модель процесса выявления информации о запуске программного обеспечения, используемая при расследовании инцидентов информационной безопасности. Входными данными предлагаемой модели являются кортежи, содержащие интересующие специалиста признаки, связанные с запуском программы: имя и полный путь до файла, время запуска и источник информации о запуске. В статье кратко описаны основные массивы данных, в которых содержится необходимая информация о запуске программы. Предлагаемая модель может быть расширена при появлении новых массивов данных, а также использована в качестве основы для создания средства автоматизации сбора и анализа информации, что позволит специалисту, проводящему расследование инцидента информационной безопасности, ускорить процесс выявления и ликвидации последствий инцидента.

Ключевые слова: *расследование инцидентов информационной безопасности, программа, следы запуска, сеть Петри, Windows.*

SOFTWARE LAUNCH ARTIFACTS IDENTIFICATION PROCESS MODEL ON WINDOWS OPERATING SYSTEMS USED IN INFORMATION SECURITY INCIDENTS INVESTIGATION

The article presents a software launch facts detecting process model used in an information security incidents investigation. The model is based on the mathematical apparatus of Petri nets, which have proven themselves in the description of complex systems in which synchronous and asynchronous, serial and parallel processes can be performed. The input data of the proposed model are tuples containing features related to the fact of launching the program: the name and full path to the file, the launch time and the source of information about the launch. The article briefly describes the main data arrays that contain the necessary information about starting the program. The proposed model can be expanded when new data arrays appear, and can also be used as the basis for creating a tool for automating information collection and analysis, which will allow a specialist conducting an information security incident investigation to speed up the process of identifying and eliminating the consequences of an incident.

Keywords: i*nformation security incident response, software, artifacts, Petri net, Windows.*

Введение

Одним из важных направлений расследования инцидента информационной безопасности (РИИБ) является выявление фактов запуска программного обеспечения (ПО) в целях как выявления несанкционированного использования программ, так и построения последовательности событий в рамках инцидента.

Согласно указу Президента Российской Федерации [1], до 1 января 2025 года на объектах критической информационной инфраструктуры (КИИ) необходимо отказаться от использования иностранного ПО, в т.ч. от операционных систем (ОС) Windows. Однако на организации, не являющиеся объектами КИИ, требования данного указа не распространяются, в связи с чем выявление следов пользовательской активности в части использования ПО в ОС Windows по-прежнему является актуальной задачей в ходе РИИБ.

В работе [2] автор обобщил перечень массивов данных, по информации из которых специалист, проводящий РИИБ, может сделать выводы о фактах запуска программ:

- Prefetch [3, 4];
- Shimcache [5];
- Amcache [6-9];
- UserAssist [10-12];
- System Resource Usage Monitor (SRUM) [13, 14].

Информация, хранящаяся в указанных массивах данных, структурно неоднородна, что затрудняет ее быстрый анализ. Также представленный перечень способствует лишь определению направления поиска интересующих сведений и не позволяет единообразно представить специалисту данные из массивов для анализа.

В настоящем исследовании автором предлагается сформировать унифицированный подход к анализу интересующих специалиста сведений с использованием модели процесса выявления информации о запуске

ПО. Указанная модель может лечь в основу средства автоматизации сбора и анализа данных, что позволит упростить и ускорить работу специалиста в рамках РИИБ.

1. Информация, хранящаяся в исследуемых массивах данных

Описание подробных структур указанных массивов данных не является целью настоящего исследования. Далее будут рассмотрены примеры сведений, которые может проанализировать специалист в рамках РИИБ.

Массив Prefetch [3] представляет собой каталог, который содержит, в том числе, файлы с расширением «pf». Они создаются отдельно для каждой программы после ее первого запуска. В каждом файле специалист может обнаружить несколько значимых для РИИБ полей, в частности имя файла программы и время ее запуска (в Windows 7 только 1 отметка времени запуска, начиная с Windows 8 сохраняется до 8 отметок). Пример содержимого файла с расширением «pf» изображен на рис. 1, где представлены наименование программы в кодировке UTF-16LE (выделено синим цветом) и 4 временные отметки (ВО) запуска в формате FILETIME (выделены красным цветом).

Массив ShimCache [5] является компонентом базы данных совместимости приложений и представляет собой двоичное значение параметра реестра AppCompatCache, расположенного в ветке HKEY_LOCAL_MACHINE\ SYSTEM\ControlSet<NUM>\Control\ Session Manager\AppCompat Cache, где <NUM> – номер ветки реестра с набором параметров системы. AppCompatCache содержит набор записей, пример которого представлен на рис. 2.

На рис. 2 полный путь до файла программы (с использованием переменной среды SYSVOL, обозначающей букву диска системного раздела) в кодировке UTF-16LE выделен синим цветом, а ВО запуска в формате FILETIME – красным.

Массив AmCache [6, 7], расположенный в файле AmCache.hve, начиная с Windows 8, является «кустом» реестра, в ветках которого

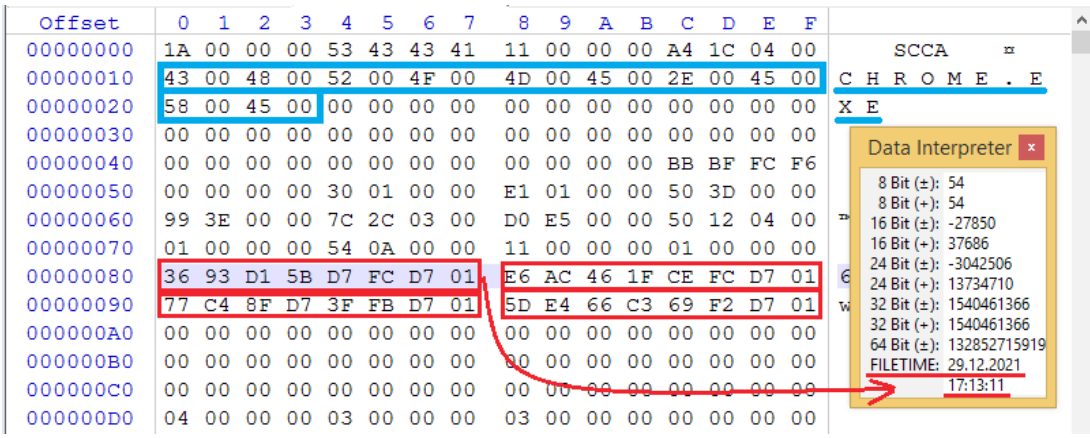

Рис. 1. Пример «pf» файла с пояснениями

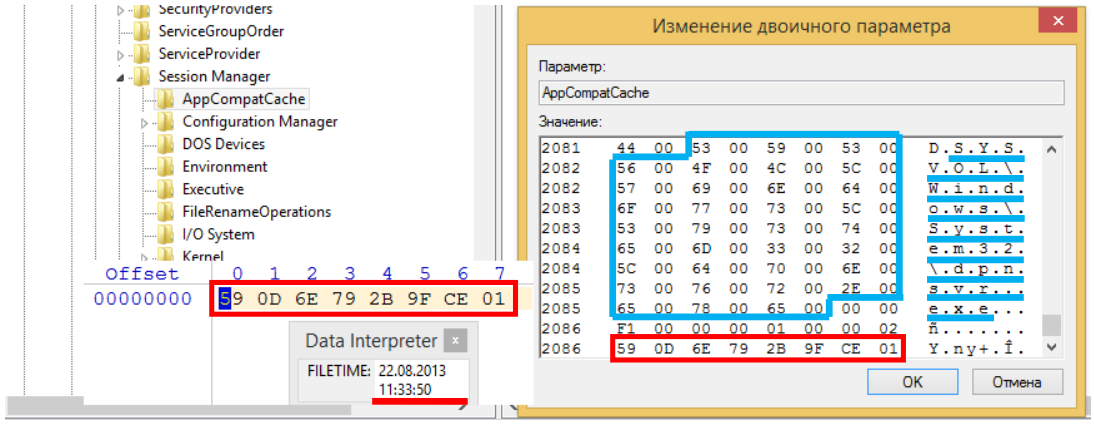

Рис. 2. Пример информации из записи параметра реестра AppCompatCache

можно обнаружить информацию о полном пути до файла программы, ВО запуска, SHA1 хэш-сумму и др. Согласно информации из [9], в зависимости от версии и релиза ОС Windows структура «куста» и наименования веток могут меняться, и эту особенность необходимо учитывать при извлечении данных из описываемого массива. Пример информации изображен на рис. 3. Синим цветом выделен полный путь до файла программы, красным – ВО в формате FILETIME.

Массив UserAssist [10-12] представляет

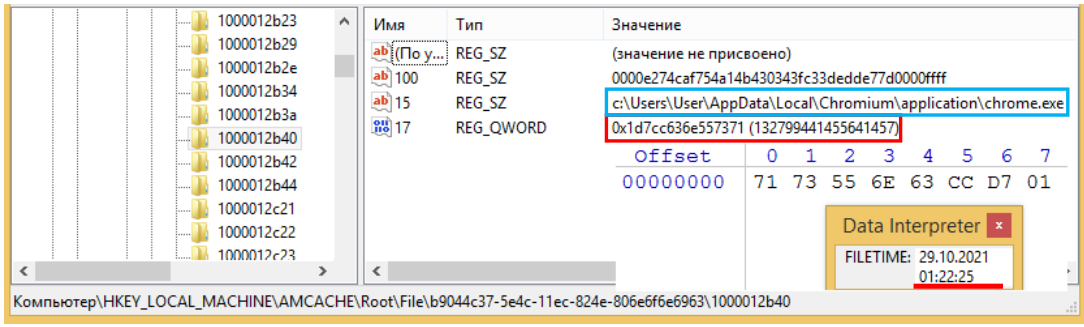

Рис. 3. Пример информации из параметров куста реестра AmCache.hve

собой набор веток реестра HKEY_USERS\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{CEBFF5CD-ACE2-4F4F-9178-9926F41749EA}\ Count, существующих для каждого пользователя исследуемой ОС. Названия параметров указанных веток содержат наименования и полные пути до файлов программ, закодированные алгоритмом ROT13 (циклический сдвиг в английском алфавите с шагом 13), а значения параметров – количество запусков программы и ВО последнего запуска. Пример информации из UserAssist представлен на рис. 4.

Массив SRUM [13, 14] представляет собой базу данных SRUDB.dat формата ESEDB [15], расположенную в каталоге %windir%\System32\sru\SRUDB.dat, где %windir% – каталог ОС (по-умолчанию «C:\Windows»). SRUM используется в ОС Windows, начиная с версии 8. База данных хранит различную информацию: имена и полные пути до файлов программ, время запуска, процессорное время в фоновом и активном режимах, объем используемой памяти, используемые сетевые соединения и пр. Пример извлекаемой из SRUDB.dat информации представлен на рис. 5.

На рис. 5 красным цветом выделен полный путь до файла программы, синим – время создания записи в SRUDB.dat, зеленым – идентификатор пользователя, который запустил программу, и его имя.

2. Формализация хранимой в исследуемых массивах информации

Для проведения экспресс-анализа в рам-

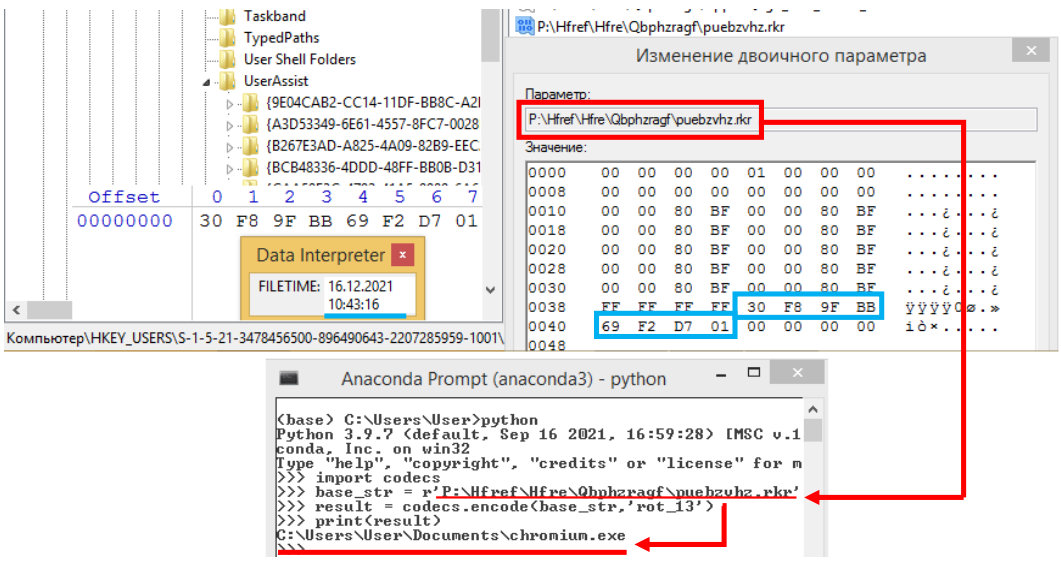

Рис. 4. Пример информации из параметров ветки реестра UserAssist

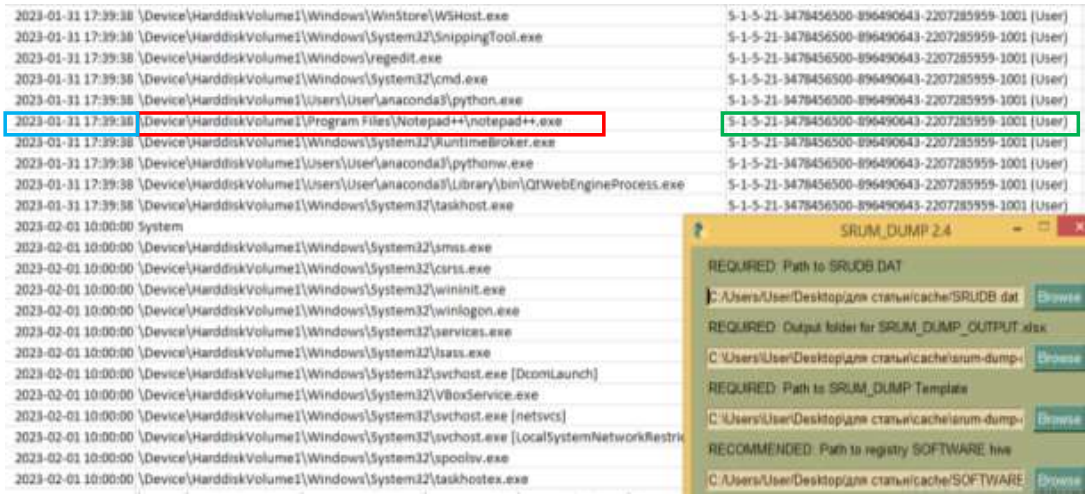

Рис. 5. Пример информации из SRUDB.dat

ках РИИБ необходимо определить перечень использованного ПО: время запуска и полные пути / имена файлов. Каждый из перечисленных выше массивов данных содержит в себе информацию, которая может быть формализована в виде кортежа r:

$$r = <N, F, S>, \quad (1)$$

где:

• N – имя или полный путь до файла программы;

• F – время запуска;

• S – массив данных из указанных выше, содержащий информацию о запуске ПО.

Представление данных из указанных массивов в виде кортежа r позволяет унифицировано хранить и обрабатывать информацию в процессе экспресс-анализа в рамках РИИБ. При необходимости углубленного анализа, например, сравнения значения хэш-суммы из AmCache с базой образцов вредоносного ПО, специалист может получить данные из первоисточника S.

3. Модель процесса выявления информации о запуске программного обеспечения в ОС Windows

Процесс выявления информации о запуске ПО может быть описан моделью M на основе сети Петри [16]:

$$M = <P, T, E, O, \mu, w, \Delta t>, \quad (2)$$

где:

• P – конечное множество позиций сети Петри;

• T – конечное множество переходов;

• E – входная функция;

• O – выходная функция;

• μ – маркировка сети фишками;

• w – «вес активации» перехода T_{20} (все остальные веса равны 1);

• Δt – временной интервал объединения записей.

Представленная сеть обладает следующими особенностями:

1. В конечное множество позиций $P = P_r \cup \cup P_{add} \cup P_{algo}$ включены: P_r ($P_0 – P_{19}$) – позиции, связанные со значениями кортежа r (1), P_{algo} ($P_{20} – P_{29}$, $P_{40} – P_{42}$) – позиции, описывающие алгоритмические условия и результат выполнения сети, P_{add} ($P_{30} – P_{39}$) – вспомогательные позиции, необходимые для корректного функционирования сети.

2. В конечное множество переходов $T = = T_r \cup T_{add} \cup T_{algo}$ включены: T_r ($T_0 – T_4$) – переходы, связанные со значениями кортежа r (1), T_{algo} ($T_{15} – T_{31}$) – переходы, описывающие алгоритмические условия и результат выполнения сети, T_{add} ($T_5 – T_{14}$) – вспомогательные позиции, необходимые для корректного функционирования сети.

3. $||\mu|| = ||P' = (P_0 – P_{14}, P_{15} – P_{29})||$.

4. Значение w соответствует половине суммы фишек в позициях $P_{20} – P_{29}$, округленной до ближайшего целого «вниз».

Элемент Δt является переменным параметром, предназначенным для упрощения восприятия информации из разных массивов данных специалистом, проводящим РИИБ. Например, если в различных массивах информация о запуске программы jigsaw.exe зафиксирована с интервалом ~3-4 секунды, то имеет смысл задать значение Δt равным 4 и вместо формирования нового кортежа r (1) дополнить значение параметра S новым массивом данных. Согласно выводам в [2] значение Δt может быть установлено в интервале от 3 до 10 секунд в силу осо-

бенностей работы системных служб, формирующих вышеуказанные массивы данных. Сравнение происходит с минимальным значением F записей, поступивших на вход сети.

Фишками маркируются только позиции P' в зависимости от наличия массивов данных. Результатом выполнения сети является одна или несколько записей, собранных в накопитель P_{42} из ранее рассмотренных массивов данных. При выполнении сети в цикле осуществляется перебор всех записей массивов.

При неверной совокупности заданных фишек в позициях P' выполнение сети прервется раньше, чем будет достигнута P_{42}. В таком случае необходимо проверить значения заданных параметров. Корректность выполнения сети апробирована в ПО PIPE [17, 18].

Примеры маркировок, обеспечивающих формирование множества R записей о запуске ПО, представлены в табл. 1.

Указанная модель M (без маркировки) представлена на рис. 6.

Таблица 1

Примеры маркировок сети

№	Маркировка	Описание	Результат
1	$P_0 – P_2 = 0$, $P_3 – P_{14} = 1$, $P_{15} = 1$, $P_{16} – P_{19} = 0$, $P_{20} = 0$, $P_{21} = 0$, $P_{22} – P_{25} = 1$, $P_{26} = 0$, $P_{27} = 1$, $P_{28} = 0$, $P_{29} = 0$	Запись Prefetch отсутствует, значение F записи из массива UserAssist минимальное, значение F записи ShimCache попадает в интервал объединения Δt, имена N совпали у записей из UserAssist и ShimCache	В накопителе формируется 3 записи в виде кортежей r: UserAssist+ShimCache, AmCache, SRUM
2	$P_0 – P_{14} = 1$, $P_{15} – P_{19} = 0$, $P_{20} – P_{29} = 1$	Записи во всех массивах попали на вход сети, значение F записи из массива UserAssist минимальное, значения F записей из остальных массивов попадают в интервал объединения Δt, значения N совпадают	В накопителе формируется 1 запись в виде кортежа r: Prefetch + UserAssist + ShimCache + AmCache + SRUM
3	$P_0 – P_5 = 0$, $P_6 – P_{14} = 1$, $P_{15} = 1$, $P_{16} = 1$, $P_{17} – P_{19} = 0$, $P_{20} – P_{23} = 0$, $P_{24} – P_{29} = 1$	Записи из Prefetch и UserAssist отсутствуют, значение F записи из массива SRUM минимальное, значения F записей из остальных массивов попадают в интервал объединения Δt, значения N совпадают	В накопителе формируется 1 запись в виде кортежа r: ShimCache + AmCache + SRUM

Подробные описания позиций и переходов приведены в табл. 2 и 3.

Пример выполнения сети с маркировкой 1 табл. 1 представлен на рис. 7 и 8. Из рисунков видно, что половина суммы фишек в позициях $P_{20} – P_{29}$, округленная «вниз» до ближайшего целого, равна 2, соответственно значение w = 2.

Таким образом, процесс формирования множества $R = \{r_1', \ldots, r_k' | k = \overline{1,K}\}$, где r_k' – k-ый кортеж r(1), обработанный моделью M(2), K – количество записей в указанных выше массивах данных, является цикличным выполнением представленной сети.

При появлении нового массива данных модель может быть расширена путем добавления аналогичных звеньев по примеру существующих массивов данных (табл. 2 и 3).

Заключение

Предлагаемая модель позволяет сформировать унифицированный подход к анализу интересующих специалиста сведений о запуске ПО. Сети Петри являются удобным и наглядным инструментом описания параллельных процессов и могут быть использованы для разработки средства автоматизации анализа информации, хранящейся в массивах данных ОС Windows. В силу растущего объема автоматизация позволит сократить временные затраты специалиста, проводящего РИИБ.

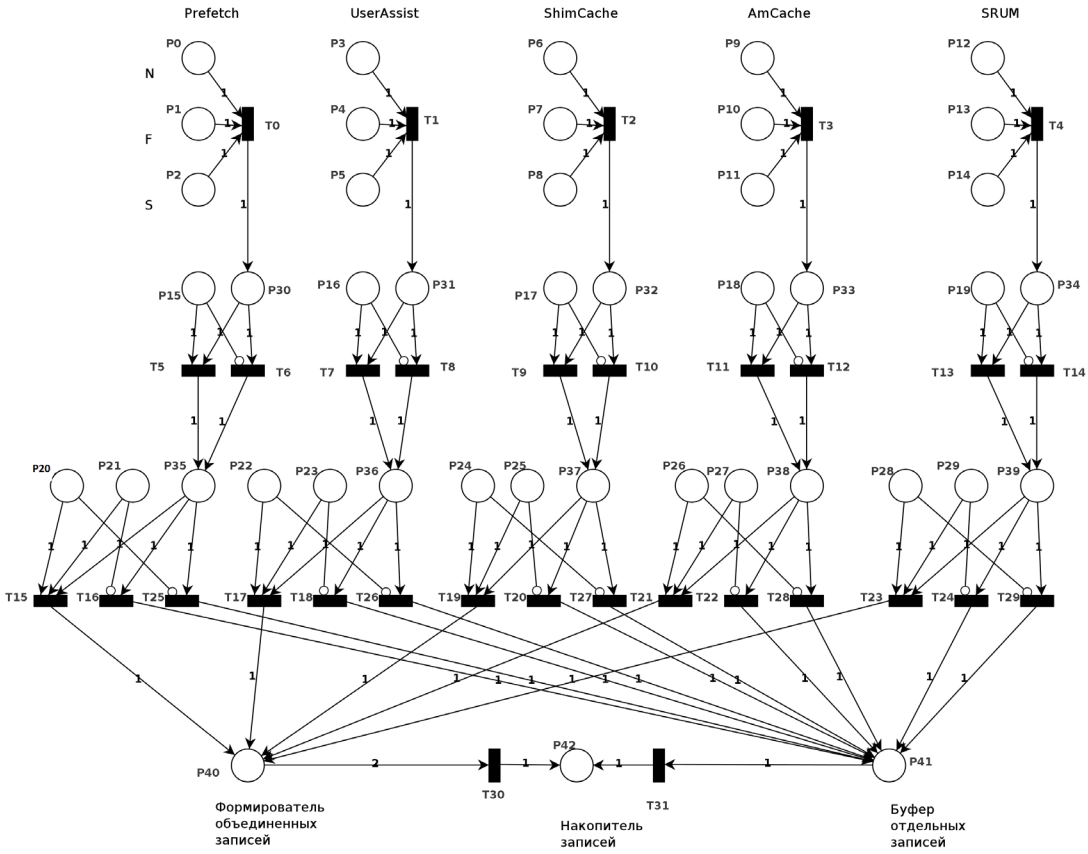

Рис. 6. Модель процесса выявления информации о запуске ПО (без маркировки)

Таблица 2

Описание позиций

Номер позиции	Описание
Позиции маркировки, определяющие начальное состояние сети Петри	
P_0	Элемент S записи Prefetch инициализирован
P_1	Элемент F записи Prefetch инициализирован
P_2	Элемент N записи Prefetch инициализирован
P_3	Элемент S записи UserAssist инициализирован
P_4	Элемент F записи UserAssist инициализирован
P_5	Элемент N записи UserAssist инициализирован
P_6	Элемент S записи ShimCache инициализирован
P_7	Элемент F записи ShimCache инициализирован
P_8	Элемент N записи ShimCache инициализирован
P_9	Элемент S записи AmCache инициализирован
P_{10}	Элемент F записи AmCache инициализирован
P_{11}	Элемент N записи AmCache инициализирован
P_{12}	Элемент S записи SRUM инициализирован
P_{13}	Элемент F записи SRUM инициализирован

Продолжение табл. 2

Номер позиции	Описание
P_{14}	Элемент N записи SRUM инициализирован
P_{15}	Запись Prefetch отсутствует
P_{16}	Запись UserAssist отсутствует
P_{17}	Запись ShimCache отсутствует
P_{18}	Запись AmCache отсутствует
P_{19}	Запись SRUM отсутствует
P_{20}	Запись Prefetch находится в интервале объединения Δt
P_{21}	Имя файла в записи Prefetch совпадает с остальными записями на входе сети
P_{22}	Запись UserAssist находится в интервале объединения Δt
P_{23}	Имя файла в записи UserAssist совпадает с остальными записями на входе сети
P_{24}	Запись ShimCache находится в интервале объединения Δt
P_{25}	Имя файла в записи ShimCache совпадает с остальными записями на входе сети
P_{26}	Запись AmCache находится в интервале объединения Δt
P_{27}	Имя файла в записи AmCache совпадает с остальными записями на входе сети
P_{28}	Запись SRUM находится в интервале объединения Δt
P_{29}	Имя файла в записи SRUM совпадает с остальными записями на входе сети
Вспомогательные и алгоритмические позиции	
P_{30}	Кортеж r записи Prefetch инициализирован
P_{31}	Кортеж r записи UserAssist инициализирован
P_{32}	Кортеж r записи ShimCache инициализирован
P_{33}	Кортеж r записи AmCache инициализирован
P_{34}	Кортеж r записи SRUM инициализирован
P_{35}	Проверка совпадения значения N и нахождения значения F в интервале объединения Δt для записи Prefetch
P_{36}	Проверка совпадения значения N и нахождения значения F в интервале объединения Δt для записи UserAssist
P_{37}	Проверка совпадения значения N и нахождения значения F в интервале объединения Δt для записи ShimCache
P_{38}	Проверка совпадения значения N и нахождения значения F в интервале объединения Δt для записи AmCache
P_{39}	Проверка совпадения значения N и нахождения значения F в интервале объединения Δt для записи SRUM
P_{40}	Накопитель записей, подлежащих объединению по элементу S
P_{41}	Накопитель самостоятельных записей, не попавших в интервал объединения Δt
P_{42}	Накопитель записей в рамках одной итерации цикла обработки массивов данных

Таблица 2

Описание переходов

Номер перехода	Описание
T_0	Инициализация кортежа r записи Prefetch
T_1	Инициализация кортежа r записи UserAssist
T_2	Инициализация кортежа r записи ShimCache
T_3	Инициализация кортежа r записи AmCache
T_4	Инициализация кортежа r записи SRUM
T_5	Выполнение условия — запись Prefetch отсутствует
T_6	Выполнение условия — запись Prefetch присутствует
T_7	Выполнение условия — запись UserAssist отсутствует
T_8	Выполнение условия — запись UserAssist присутствует
T_9	Выполнение условия — запись ShimCache отсутствует
T_{10}	Выполнение условия — запись ShimCache присутствует
T_{11}	Выполнение условия — запись AmCache отсутствует
T_{12}	Выполнение условия — запись AmCache присутствует
T_{13}	Выполнение условия — запись SRUM отсутствует
T_{14}	Выполнение условия — запись SRUM присутствует
T_{15}	Выполнение условия — запись Prefetch находится в интервале объединения Δt и имя файла совпадает с записями из остальных массивов
T_{16}	Выполнение условия — запись Prefetch не находится в интервале объединения Δt
T_{25}	Выполнение условия — имя файла записи Prefetch не совпадает с записями из остальных массивов
T_{17}	Выполнение условия — запись UserAssist находится в интервале объединения Δt и имя файла совпадает с записями из остальных массивов
T_{18}	Выполнение условия — запись UserAssist не находится в интервале объединения Δt
T_{26}	Выполнение условия — имя файла записи UserAssist не совпадает с записями из остальных массивов
T_{19}	Выполнение условия — запись ShimCache находится в интервале объединения Δt и имя файла совпадает с записями из остальных массивов
T_{20}	Выполнение условия — запись ShimCache не находится в интервале объединения Δt
T_{27}	Выполнение условия — имя файла записи ShimCache не совпадает с записями из остальных массивов
T_{21}	Выполнение условия — запись AmCache находится в интервале объединения Δt и имя файла совпадает с записями из остальных массивов
T_{22}	Выполнение условия — запись AmCache не находится в интервале объединения Δt
T_{28}	Выполнение условия — имя файла записи AmCache не совпадает с записями из остальных массивов
T_{23}	Выполнение условия — запись SRUM находится в интервале объединения Δt и имя файла совпадает с записями из остальных массивов
T_{24}	Выполнение условия — запись SRUM не находится в интервале объединения Δt
T_{29}	Выполнение условия — имя файла записи SRUM не совпадает с записями из остальных массивов
T_{30}	Перенос объединенной записи в результирующий накопитель
T_{31}	Перенос раздельных записей в результирующий накопитель

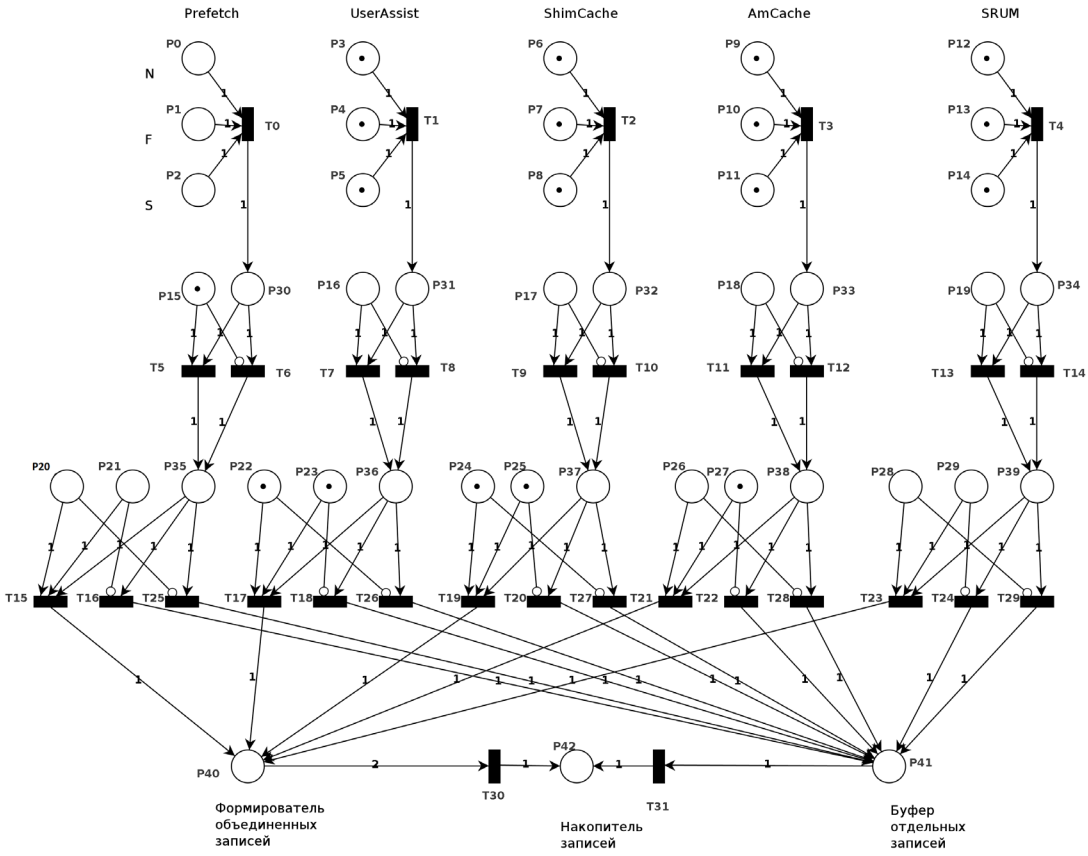

Рис. 7. Задание маркировки модели по параметрам 1 табл. 1

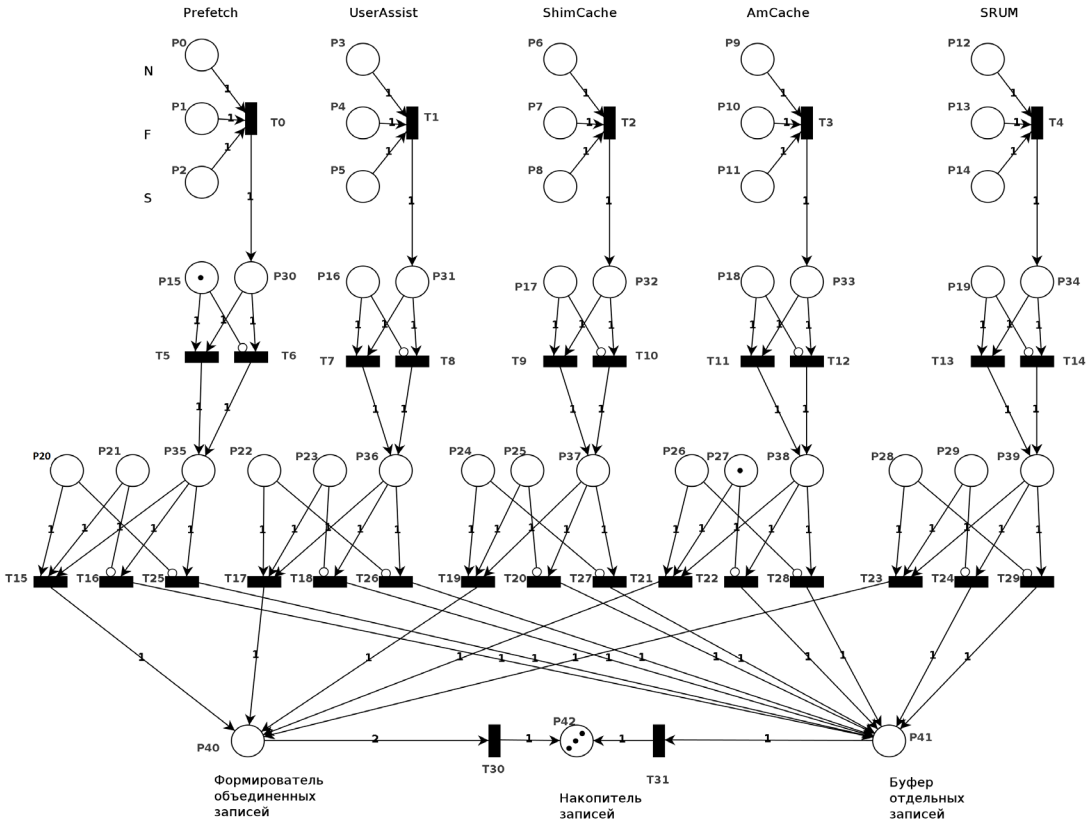

Рис. 8. Выполненная сеть Петри с обработанными тремя записями в накопителе

Литература

1. Указ Президента Российской Федерации от 30.03.2022 № 166 «О мерах по обеспечению технологической независимости и безопасности критической информационной инфраструктуры Российской Федерации».

2. Battle of the Shims. [Электронный ресурс]. URL: https://svch0st.medium.com/battle-of-the-shims-60fdae38264e (дата обращения: 31.01.2023).

3. libscca/Windows Prefetch File (PF) format.asciidoc at main — libyal/libscca — GitHub. [Электронный ресурс]. URL: https://github.com/libyal/libscca/blob/main/documentation/Windows Prefetch File (PF) format.asciidoc (дата обращения: 31.01.2023).

4. GitHub — EricZimmerman/Prefetch [Электронный ресурс]. URL: https://github.com/EricZimmerman/Prefetch (дата обращения: 31.01.2023).

5. ShimCacheParser. [Электронный ресурс]. URL: https://github.com/mandiant/ShimCacheParser (дата обращения: 31.01.2023).

6. Singh B., Singh U. Leveraging the Windows Amcache.hve File in Forensic Investigations // Journal of Digital Forensics, Security and Law, 2016. Vol. 11. No. 4. pp. 37-54.

7. Amcache.hve in Windows 8 — Goldmine for malware hunters. [Электронный ресурс]. URL: https://www.swiftforensics.com/2013/12/amcachehve-in-windows-8-goldmine-for.html (дата обращения: 31.01.2023).

8. Journey into Incident Response: Revealing the RecentFileCache.bcf File. [Электронный ресурс]. URL: https://journeyintoir.blogspot.com/2013/revealing-recentfilecachebcf-file.html (дата обращения: 31.01.2023).

9. Analysis of the AmCache V2. [Электронный ресурс]. URL: https://www.ssi.gouv.fr/uploads/2019/01/anssi-coriin_2019-analysis_amcache.pdf (дата обращения: 31.01.2023).

10. Singh B., Singh U. Program Execution Analysis using UserAssist Key in Modern Windows / Proceedings of the 14th International Joint Conference on e-Business and Telecommunications (ICETE 2017). Vol. 4. pp. 420-429.

11. User Assist key — Windows Registry knowledge base. [Электронный ресурс]. URL: https://winreg-kb.readthedocs.io/en/latest/sources/explorer-keys/User-assist.html (дата обращения: 31.01.2023).

12. UserAssist — with a pinch of Salt — As an "Evidence of Execution". [Электронный ресурс]. URL: https://imphash.medium.com/userassist-with-a-pinch-of-salt-as-an-evidence-of-execution (дата обращения: 31.01.2023).

13. Khatri Y. Forensic implications of System Resource Usage Monitor (SRUM) data in Windows 8 / Digital Investigation, 2015. Vol. 12. pp. 53-65.

14. GitHub — MarkBaggett/srum-dump [Электронный ресурс]. URL: https://github.com/MarkBaggett/srum-dump (дата обращения: 31.01.2023).

15. Microsoft. Extensible storage engine. [Электронный ресурс]. URL: http://msdn.microsoft.com/en-us/library/office/gg269259(v=exchg.10).aspx (дата обращения: 31.01.2023).

16. Советов Б.Я., Яковлев С.А. Моделирование систем: Учеб. для вузов — 3-е юд., перераб. и доп. — М.: Высш. шк., 2001. — 343 с: ил.

17. GitHub — sarahtattersall/PIPE: PIPE – Platform Independent Petri Net Editor. [Электронный ресурс]. URL: https://github.com/sarahtattersall/PIPE (дата обращения: 31.01.2023).

18. Dingle N., Knottenbelt W., Suto T. PIPE2: A tool for the Performance Evaluation of Generalized Stochastic Petri Nets // ACM SIGMETRICS Performance Evaluation Review (Special Issue on Tools for Computer Performance Modeling and Reliability Analysis), 2009. № 36. pp. 34–39.

References

1. Ukaz Prezidenta Rossiyskoy Federatsii ot 30.03.2022 № 166 «O merakh po obespecheniyu tekhnologicheskoy nezavisimosti i bezopasnosti kriticheskoy informatsionnoy infrastruktury Rossiyskoy Federatsii».

2. Battle of the Shims. [Elektronnyy resurs]. URL: https://svch0st.medium.com/battle-of-the-shims-60fdae38264e (data obrashcheniya: 31.01.2023).

3. libscca/Windows Prefetch File (PF) format.asciidoc at main — libyal/libscca — GitHub. [Elektronnyy resurs]. URL: https://github.com/libyal/libscca/blob/main/documentation/Windows Prefetch File (PF) format.asciidoc (data obrashcheniya: 31.01.2023).

4. GitHub — EricZimmerman/Prefetch [Elektronnyy resurs]. URL: https://github.com/EricZimmerman/Prefetch (data obrashcheniya: 31.01.2023).

5. ShimCacheParser. [Elektronnyy resurs]. URL: https://github.com/mandiant/ShimCacheParser (data obrashcheniya: 31.01.2023).

6. Singh B., Singh U. Leveraging the Windows Amcache.hve File in Forensic Investigations // Journal of Digital Forensics, Security and Law, 2016. Vol. 11. No. 4. pp. 37-54.

7. Amcache.hve in Windows 8 — Goldmine for malware hunters. [Elektronnyy resurs]. URL: https://www.swiftforensics.com/2013/12/amcachehve-in-windows-8-goldmine-for.html (data obrashcheniya: 31.01.2023).

8. Journey into Incident Response: Revealing the RecentFileCache.bcf File. [Elektronnyy resurs]. URL: https://journeyintoir.blogspot.com/2013/revealing-recentfilecachebcf-file.html (data obrashcheniya: 31.01.2023).

9. Analysis of the AmCache V2. [Elektronnyy resurs]. URL: https://www.ssi.gouv.fr/uploads/2019/01/anssi-coriin_2019-analysis_amcache.pdf (data obrashcheniya: 31.01.2023).

10. Singh B., Singh U. Program Execution Analysis using UserAssist Key in Modern Windows / Proceedings of the 14th International Joint Conference on e-Business and Telecommunications (ICETE 2017). Vol. 4. pp. 420-429.

11. User Assist key — Windows Registry knowledge base. [Elektronnyy resurs]. URL: https://winreg-kb.readthedocs.io/en/latest/sources/explorer-keys/User-assist.html (data obrashcheniya: 31.01.2023).

12. UserAssist — with a pinch of Salt — As an "Evidence of Execution". [Elektronnyy resurs]. URL: https://imphash.medium.com/userassist-with-a-pinch-of-salt-as-an-evidence-of-execution (data obrashcheniya: 31.01.2023).

13. Khatri Y. Forensic implications of System Resource Usage Monitor (SRUM) data in Windows 8 / Digital Investigation, 2015. Vol. 12. pp. 53-65.

14. GitHub — MarkBaggett/srum-dump [Elektronnyy resurs]. URL: https://github.com/MarkBaggett/srum-dump (data obrashcheniya: 31.01.2023).

15. Microsoft. Extensible storage engine. [Elektronnyy resurs]. URL: http://msdn.microsoft.com/en-us/library/office/gg269259(v=exchg.10).aspx (data obrashcheniya: 31.01.2023).

16. Sovetov B.Ya., Yakovlev S.A. Modelirovanie sistem: Ucheb. dlya vuzov — 3-e yud., pererab. i dop. — M.: Vyssh. shk., 2001. — 343 s: il.

17. GitHub — sarahtattersall/PIPE: PIPE – Platform Independent Petri Net Editor. [Elektronnyy resurs]. URL: https://github.com/sarahtattersall/PIPE (data obrashcheniya: 31.01.2023).

18. Dingle N., Knottenbelt W., Suto T. PIPE2: A tool for the Performance Evaluation of Generalized Stochastic Petri Nets // ACM SIGMETRICS Performance Evaluation Review (Special Issue on Tools for Computer Performance Modeling and Reliability Analysis), 2009. № 36. pp. 34–39.

ГИБИЛИНДА Роман Владимирович, кандидат технических наук, доцент учебно-научного центра «Информационная безопасность» ИРИТРтФ УрФУ им. первого Президента России Б.Н. Ельцина. 620002, г. Екатеринбург, ул. Мира, 19. E-mail: r.v.gibilinda@urfu.ru

GIBILINDA Roman Vladimirovich, candidate of technical sciences, Associate Professor of Educational and Scientific Center "Information Security", Institute of Radio electronics and Information Technologies, Ural Federal University named after first President of Russia B.N. Yeltsin. 620002, Yekaterinburg, Mira str., 19. E-mail: r.v.gibilinda@urfu.ru