Исследование методов обнаружения вредоносного исполнимого кода в высокоскоростных каналах передачи данных

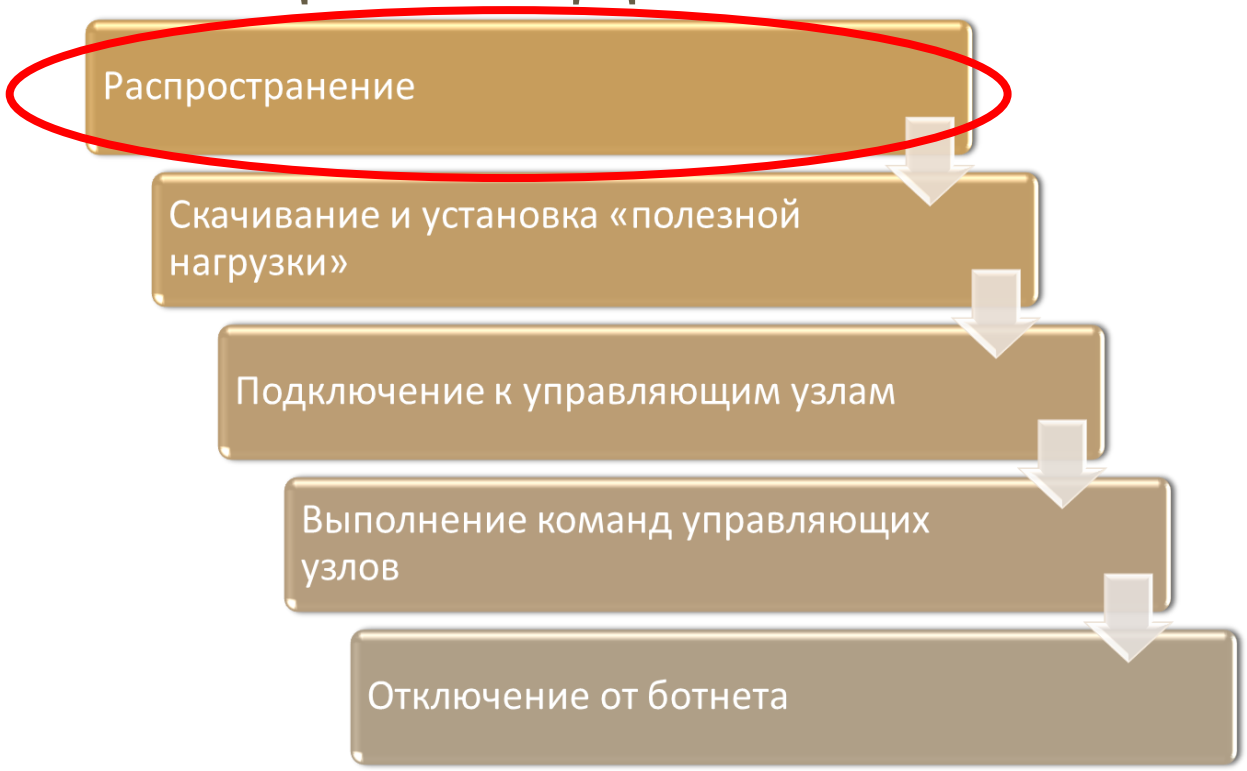
Гайворонская Светлана

аспирантка 1 года ВМК МГУ им. М.В. Ломоносова

Научный руководитель:

к.ф.-м.н. Гамаюнов Д.Ю.

Тема работы в рамках более общей задачи

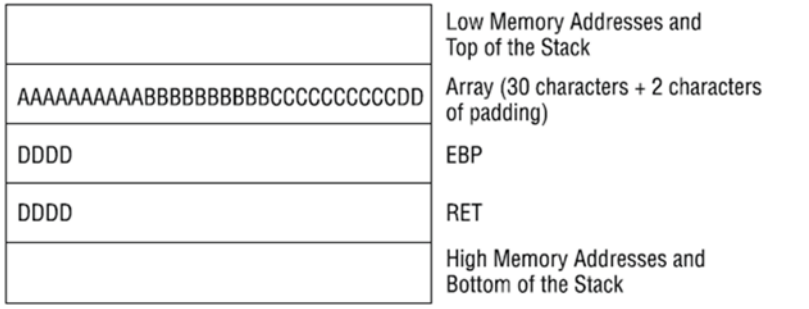


Рассматривается механизм распространения с помощью сетевых червей через уязвимости в распространенном ПО.

mailto: sadie@lvk.cs.msu.su

Шеллкод

– набор машинных инструкций, осуществляющих эксплуатирование удаленной уязвимости.



mailto: sadie@lvk.cs.msu.su

Классификация шеллкодов

Пространство шеллкодов разбивается на классы $K_1 \ldots K_l$ по набору признаков вредоносных инструкций.

mailto: sadie@lvk.cs.msu.su

Классификация методов обнаружения шеллкодов

- Статические;
- Динамические;
- Абстрактная интерпретация - оценка изменений кода программы и достижимости отдельных ее блоков без реального выполнения;
- Гибридные.

mailto: sadie@lvk.cs.msu.su

Анализ методов

Статические методы (ButterCup, Hamsa, Polygraph, Structural analysis, Stride, Racewalk, Styx, SigFree, STILL, Semantic-aware malware detection)

Возможно полное покрытие кода программы

Быстрее динамического (~O(N))

Задача обнаружения метаморфного шеллкода неразрешима

Задача обнаружения полиморфного шеллкода NP-полна

Динамические методы (Emulation, NSC Emulation, IGPSA)

Устойчивы к обфускации

Требуют больше накладных расходов

Покрытие программы не полно

Сложность эмуляции окружения

Существуют техники обнаружения выполнения программы в виртуальном окружении

mailto: sadie@lvk.cs.msu.su

Показатели эффективности методов

- Вычислительная сложность;
- Процент ошибок второго рода. Ошибка второго рода – определение методом легитимного объекта вредоносным;
- Полнота покрытия классов обнаруживаемых шеллкодов.

mailto: sadie@lvk.cs.msu.su

Результаты обзора методов

- Ни один из существующих методов обнаружения шеллкодов не обеспечивает полного покрытия классов $K_1 \dots K_l$ шеллкодов;
- Методы, имеющие низкую вычислительную сложность, характеризуются высокой долей ложных срабатываний;
- Методы, имеющие высокую точность, характеризуются высокой вычислительной сложностью.

Актуальна разработка комбинированного метода, который позволит снизить вероятность ложных срабатываний при одновременном уменьшении вычислительной сложности по сравнению с простой комбинацией существующих методов.

mailto: sadie@lvk.cs.msu.su

Задача работы

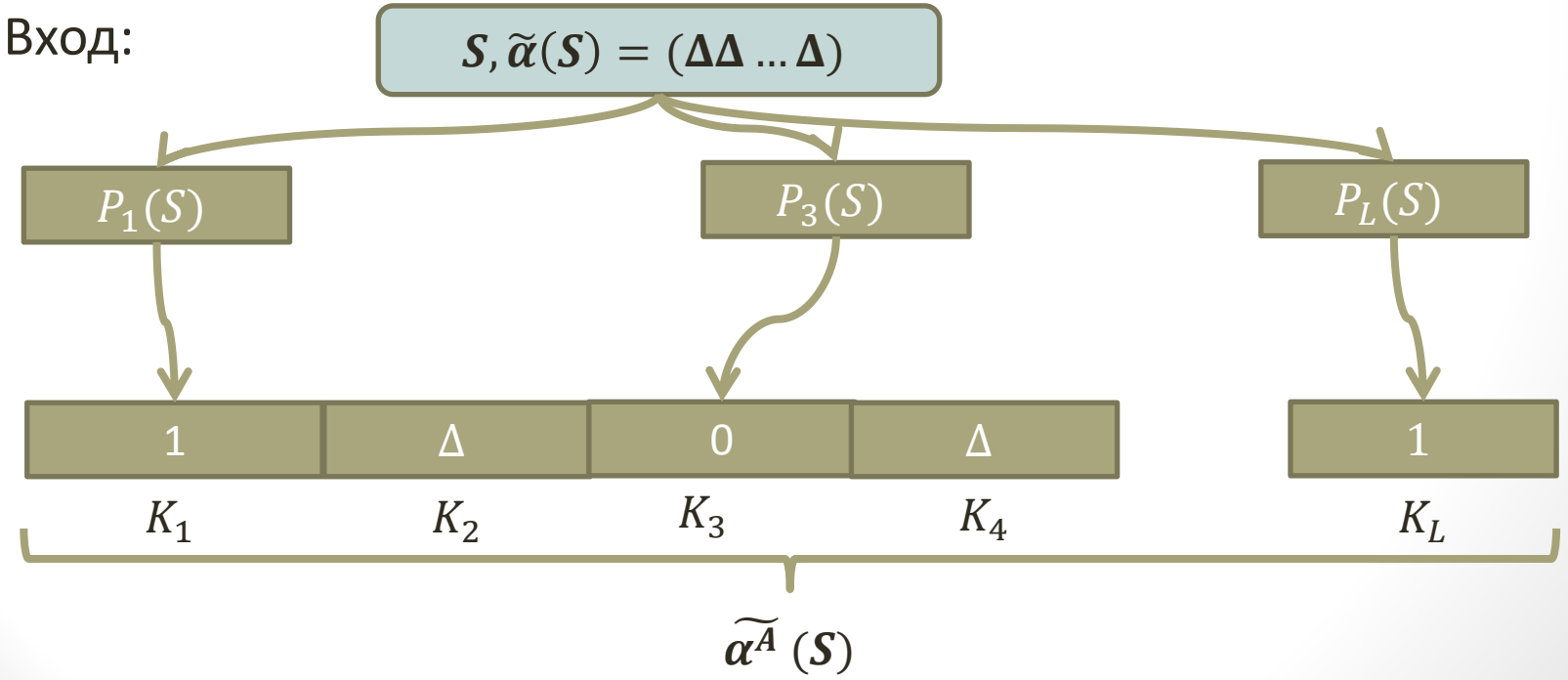
Требуется разработать комбинированный алгоритм обнаружения шеллкодов в высокоскоростных каналах такой, что:

- вероятность ложных срабатываний алгоритма минимальна;
- вычислительная сложность алгоритма минимальна по сравнению с простой комбинацией алгоритмов;
- обеспечивается полное покрытие классов шеллкодов;
- учитывается профиль трафика в канале.

mailto: sadie@lvk.cs.msu.su

Информационный вектор

- Каждому классу K_i шеллкода сопоставлен элементарный предикат $P_i(S) \in \{0, 1, \Delta\}$, определяющий принадлежность набора инструкций $S = \{I_1, \dots, I_n\}$ заданному классу шеллкодов.
- Информация о вхождении S в классы $K_1, \dots, K_l$ кодируется **информационным вектором** $\tilde{\alpha}(S) = (\alpha_1 \alpha_2 \dots \alpha_l)$.



mailto: sadie@lvk.cs.msu.su

Определения

- **Определение 1.** Набор инструкций S считается **вредоносным** (шеллкодом), если $|\tilde{\alpha}(S)| \neq 0$. Набор инструкций S считается легитимным, если $|\tilde{\alpha}(S)| = 0$.

Вычисление нормы вектора: $|\tilde{\alpha}(S)| = \sqrt{\alpha_1^2 + \cdots + \alpha_L^2}$, $\Delta^2 = 0$.

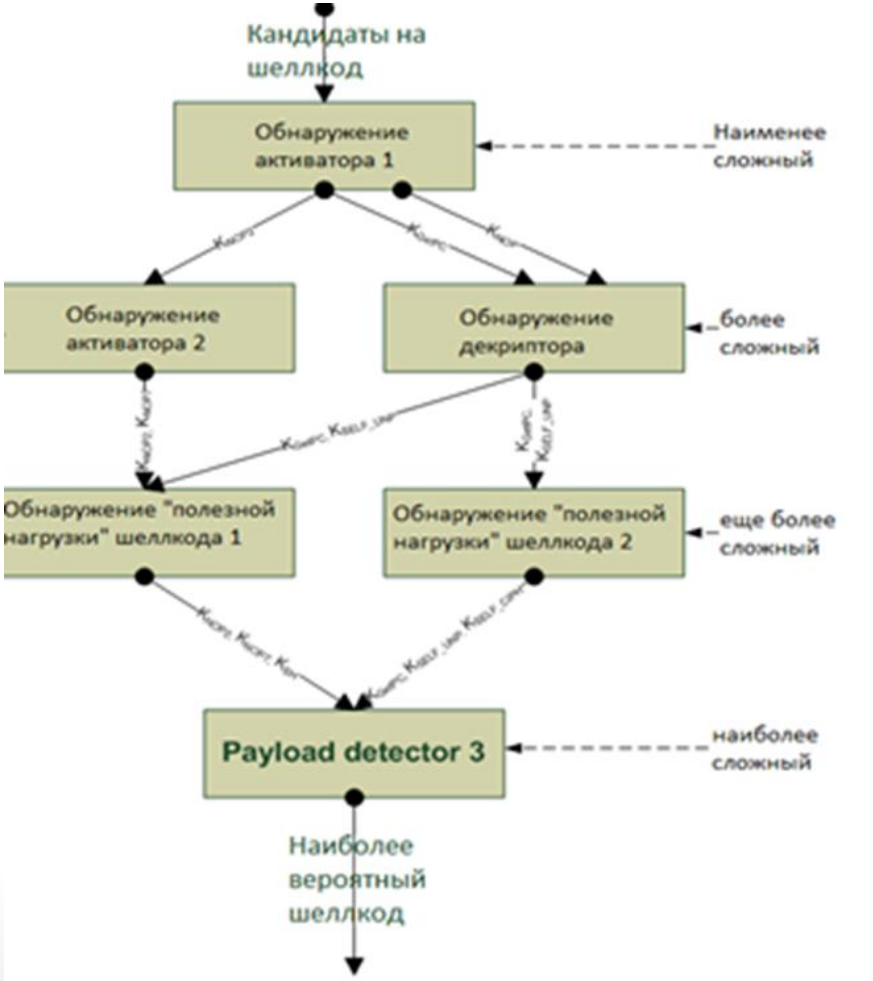
- Метод обнаружения шеллкода A рассматривается как классификатор, вычисляющий значения элементарных предикатов. На выходе имеем информационный вектор: $\widetilde{\alpha^A} = \left(\alpha_1^A \alpha_2^A \ldots \alpha_l^A\right)$.

- **Определение 2.** Вероятность ошибкой второго рода FP метода A:

$$FP(A) = \boldsymbol{P}\left(\left|\widetilde{\alpha^A}\right| \geq 1 \,\middle|\, |\tilde{\alpha}|=0\right).$$

mailto: sadie@lvk.cs.msu.su

Идея решения



Задача построения ориентированного графа G={V,E}:

- V – множество узлов-классификаторов
- E – множество дуг, представляющих из себя пути следования потока данных.

mailto: sadie@lvk.cs.msu.su

Уменьшение потоков

Если классификатор оценивает рассматриваемый объект как легитимный, то такой объект в другие классификаторы не транслируется.

mailto: sadie@lvk.cs.msu.su

Спасибо за внимание!

mailto: sadie@lvk.cs.msu.su

Используемая литература (1\3)

- Y. I. Zhuravlev, Algebraic approach to the solution of recognition or classification problems. Pattern recognition and image analysis, 1998, vol. 8; no.1, 59-100
- U. Payer, M. Lamberger, P. Teufl, Hybrid engine for polymorphic shellcode detection. In: Proceedings of the Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA05). Berlin: Springer-Verlag, 2005. 19-31
- R. Chinchani, E. Berg, A fast static analysis approach to detect exploit code inside network flows. In: Proceedings of the 8th International Symposium on Recent Advances in Intrusion Detection (RAID'05). Berlin: Springer-Verlag, 2005. 284-308
- C. Kruegel, E. Kirda, D. Mutz, et al., Polymorphic worm detection using structural information of executables. In: Proceedings of the 8th International Symposium on Recent Advances in Intrusion Detection (RAID'05). Berlin: Springer-Verlag, 2005
- P. Akritidis, E. Markatos, M. Polychronakis, and K Anagnostakis, Stride: Polymorphic sled detection through instruction sequence analysis. In Proc. of the 20th IFIP International Information Security Conference (SEC'05), 2005
- D. Gamayunov, N. T. Minh Quan, F. Sakharov, E. Toroshchin Racewalk: fast instruction frequency analysis and classification for shellcode detection in network flow In: 2009 European Conference on Computer Network Defense. Milano, Italy, 2009
- A. Pasupulati, J. Coit, K. Levitt, et al., Buttercup: On network-based detection of polymorphic buffer overflow vulnerabilities. In: Proceedings of Network Operations and Management Symposium 2004. Washington: IEEE Computer Society, 2004

mailto: sadie@lvk.cs.msu.su

Используемая литература (2\3)

- M. Weiser, Program Slicing: Formal, Psychological and Practical Investigations of an Automatic Program Abstraction Method. PhD thesis, The University of Michigan, Ann Arbor, Michigan, 1979
- X. Wang, Y. Jhi, S. Zhu, Protecting Web Services from Remote Exploit Code: A Static Analysis Approach In Proc. of the 17th International conference on World Wide Web (WWW'08), 2008.
- M. Christodorescu, S. Jha, S. A. Seshia, D. Song, and R. E. Bryant, Semantics-aware malware detection. In Proc. of 2005 IEEE Symposium on Security and Privacy (S&P'05), 2005
- M. Polychronakis, K. G. Anagnostakis, E. P. Markatos, Network-level polymorphic shellcode detection using emulation. In:Proceedings of the Conference on Detection of Intrusions and Malware & Vulnerability Assessment. Berlin: Springer-Verlag, 2006
- Q. Zhang, D. S. Reeves, P. Ning, et al., Analyzing network traffic to detect self-decrypting exploit code. In: Proceedings of the 2nd ACM Symposium on InformAtion, Computer and Communications Security, New York: ACM, 2007. 4-12
- L. Wang, H. Duan, X. Li, Dynamic emulation based modeling and detection of polymorphic shellcode at the network level Science in China Series F: Information Sciences Volume 51, Number 11, 1883-1897.

mailto: sadie@lvk.cs.msu.su

Используемая литература (3\3)

- M. Polychronakis, K. G. Anagnostakis, E. P. Markatos Emulation-based Detection of Non-self-contained Polymorphic Shellcode In Proc. of the 10th international conference on Recent advances in intrusion detection (RAID'07), 2007.
- E. Filiol, Metamorphism, formal grammars and undecidable code mutation. International Journal of Computer Science,2, 2007
- Z. Li, M. Sanghi, Y. Chen, et al., Hamsa: Fast signature generation for zero-day polymorphic worms with provable attack resilience. In: Proceedings of 2006 IEEE Symposium on Security and Privacy (S&P'06). Washington: IEEE Computer Society, 2006. 32-47
- J. Newsome, B. Karp, D. Song, Polygraph: automatically generating signatures for polymorphic worms. In: Proceedings of 2005 IEEE Symposium on Security and Privacy (S&P'05). Washington: IEEE Computer Society, 2005. 226-241
- P. Royal, M. Halpin, D. Dagon, R. Edmonds, W. Lee, PolyUnpack: Automating the Hidden-Code Extraction of Unpack-Executing Malware In: Computer Security Applications Conference (ACSAC'06), 2006.
- T. Toth, C. Kruegel, Accurate Buffer Overflow Detection via Abstract Payload Execution In Proc. of the 5th international conference on Recent advances in intrusion detection (RAID'02), 2002.
- X. Wang, C. C. Pan, P. Liu, S. Zhu, Sigfree: A signature free buffer overflow attack blocker. In 15th Usenix Security Symposium, July 2006

mailto: sadie@lvk.cs.msu.su

Классы шеллкодов. Активаторы

- K_{NOP_1} - класс объектов, содержащий простейщий NOP-след – набор инструкций 0x90;
- K_{NOP_2} - объекты, содержащие однобайтный NOP-эквивалентный след;
- K_{NOP_3} - объекты, содержащие многобайтный NOP-эквивалентный след;
- K_{NOP_4} - объекты, содержащие 4-х байтный выровненный след;
- K_{NOP_5} - объекты, содержащие батутный след;
- K_{NOP_6} - объекты, содержащие обфусцированный батутный след;
- K_{NOP_7} - объекты, содержащие след, устойчивый к статичному анализу;
- K_{GetPC} - объекты, содержащие GetPC код.

mailto: sadie@lvk.cs.msu.su

Классы шеллкодов. Декрипторы

- K_{SELF_UNP} - класс самораспаковывающихся шеллкодов;
- K_{SELF_CIPH} - класс самошифрующихся шеллкодов.

mailto: sadie@lvk.cs.msu.su

Классы шеллкодов. Полезная нагрузка

- K_{SH} - класс необфусцированных шеллкодов;
- K_{DATA} - класс шеллкодов с обфускацией данных. К примеру, ASCII символы могут быть заменены на UNICODE;
- K_{ALT_OP} - класс шеллкодов, обфусцированных путем вставки альтернативных операторов;
- K_R - класс шеллкодов, обфусцированных путем изменения порядка следования инструкций;
- K_{ALT_I} - класс шеллкодов, обфусцированных путем замены инструкций на инструкции с той же операционной семантикой;
- K_{INJ} - класс шеллкодов, обфусцированных путем вставки инструкций;
- K_{MET} - класс метаморфных шеллкодов;
- K_{NSC} (non-self-contained) – класс полиморфных шеллкодов, не базирующихся ни на одной из форм GetPC кода и не производящих чтения из памяти, входящей в адресное пространство кода во время выполнения своей декрипторной части.

mailto: sadie@lvk.cs.msu.su

Классы шеллкодов. Зона адресов возврата

- K_{RET} - класс шеллкодов, выделяемых путем обнаружения в них зоны адреса возврата;
- K_{RET_+} - класс шеллкодов, содержащих обфусцированную зону адресов возврата. Например, возможно изменение порядка младших бит адреса. В таком случае управление будет передаваться на различные позиции стека, но всегда на какую-то часть NOP-следа.

mailto: sadie@lvk.cs.msu.su