Ataki na procesory

Czy Twoje oprogramowanie jest bezpieczne? Być może. A czy Twój procesor również jest bezpieczny? Tak? Na pewno?

CZY MOIM PIENIĄDZOM W BANKU NIC NIE GROZI?

Haker – pierwsze, co przychodzi na myśl większości osób, to obraz wykreowany przez Hollywood – osoba, która w ciągu kilkunastu sekund włamie się do najpilniej strzeżonego rządowego systemu przy akompaniamencie chwytliwej muzyki i wspaniałych obrazów animacji komputerowej pojawiającej się na monitorze hakera. Co bardziej realistyczne produkcje pokazują stosowanie rzeczywistych metod wykorzystujących podatności oprogramowania. Bardziej świadomi użytkownicy komputerów wiedzą, że znajdowanie luk w oprogramowaniu jest procesem żmudnym i, wbrew pozorom, dość nudnym.

Niemniej jednak ciągle słyszymy o nowych atakach na oprogramowanie, które często wiąże się z finansowymi szkodami mierzonymi w wielocyfrowych kwotach. Dlatego na froncie twórcy oprogramowania kontra czarne kapelusze trwa nieustający wyścig zbrojeń. Twórcy oprogramowania nie ustają w wysiłkach aktualizacji swojego oprogramowania, wierząc, że akurat ta najnowsza wersja programu będzie wolna od błędów i nie pozwoli dla przykładu na żaden nieautoryzowany dostęp do informacji poufnej klientów. Przyjmijmy więc na potrzeby dalszej części tego artykułu, że ta utopijna wizja się ziści, że któregoś dnia faktycznie oprogramowanie nie będzie posiadało żadnych błędów logicznych. Czy wtedy można spać spokojnie i nie martwić się o bezpieczeństwo swoich oszczędności w banku internetowym? Okazuje się, że niekoniecznie.

ATAKI SPRZĘTOWE

Ostatnimi czasy modne stało się wyszukiwanie podatności sprzętowych. O ile dotychczas użytkownicy dopuszczali możliwość istnienia błędów w oprogramowaniu, to bardzo wielu z nich ufało bezgranicznie w jakość i bezpieczeństwo sprzętu (procesorów), na jakim działało. Niestety, istnieje coraz więcej badań pokazujących, jak bardzo było to błędne założenie. Ostatnio znacząco rozwinęła się gałąź ataków, tzw. typu *side-channel*, czyli ataków nie opierających się na bezpośrednim dostępie do wrażliwych danych. Okazało się, że istnieją sposoby odczytu wrażliwych danych, analizując skutki uboczne (ang. *side-effects*) wykonywania na nich operacji przez procesor. W tym artykule zostaną przybliżone metody, jakimi przestępca może się posłużyć, by zdobyć dostęp do danych, do których nie powinien.

TROCHĘ TEORII

Zanim przejdziemy do szczegółowego omawiania zagadnień związanych z atakami typu *side-channel*, przyjrzyjmy się skrótowo zasadzie działania współczesnych procesorów.

TROCHĘ TEORII – PIPELINE

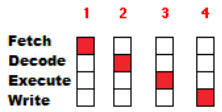
Procesor dzieli wykonanie instrukcji na kilka etapów. By odrobinę uprościć zagadnienie, przyjrzyjmy się, jak odbywało się to za dawnych czasów, czyli wtedy, gdy nie śniło się o procesorach na tyle wydajnych, by mogły pracować w telefonach komórkowych. Ba, do czasów, gdy nie śniło się nawet o samych telefonach komórkowych!

Aby procesor wykonał pojedynczą instrukcję kodu maszynowego, musi przeprowadzić szereg operacji logicznych na swoich wewnętrznych elementach wykonawczych zbudowanych z bramek logicznych. Dlatego każda instrukcja wykonywana przez jednostkę centralną musiała przejść przez następujące etapy (ang. *stage*):

- *Fetch* – kod instrukcji zostaje pobrany z pamięci,
- *Decode* – jednostka dekodująca analizuje instrukcję, by wiedzieć, co ma zrobić (zapis, dodawanie, mnożenie, skok itp.),
- *Execute* – wykonanie zdekodowanej instrukcji przez procesor,
- *Write* – zapisanie wyniku obliczeń do pamięci zewnętrznej lub rejestru procesora.

W najbardziej naiwnym podejściu etapy wykonują się jeden po drugim w rytmie zegara systemowego, który synchronizuje przetwarzanie instrukcji (rozumianej jako przepływ danych) pomiędzy blokami funkcjonalnymi procesora. Przykładem może być mały mikrokontroler z rodziny 8051 (rok 1980), który posiadał 12 etapów wykonania instrukcji. Fakt, że maksymalny zegar systemowy to 12 MHz, oznaczał, że jednostka była w stanie wykonać co najwyżej 1 milion operacji na sekundę.

Uproszczoną sytuację, w której procesor posiada tylko 4 etapy, pokazano na Rysunku 1.



Rysunek 1. Przepływ wykonania instrukcji w procesorze (inspiracją do powstania powyższej ilustracji oraz rysunków: 2, 3, 5, 6, 7, 8, 9 i 10 jest praca Davida K. Every)

Przetwarzanie takie ma wadę – jest powolne. Z biegiem czasu, żeby osiągnąć większą wydajność, wymyślono bardzo sprytny mechanizm zwany przetwarzaniem potokowym (ang. *pipeline*). Otóż zauważono, że instrukcja, wędrując przez wszystkie etapy – zawsze zajmuje tylko jeden „stage". Zatem pozostałe są wolne i nic nie robią, prawda? Wystarczy w każdym cyklu zegarowym pobierać nowe instrukcje, by wszystkie „stage" były obsadzone. Przypomina to trochę linię produkcyjną w fabryce Henrego Forda, gdzie każde stanowisko wykonuje tylko jedną czynność, a produkt, w tym przypadku przetwarzana instrukcja, przemiesza się w dół linii produkcyjnej. Pomysł prosty