Honeytokeny – obrana útokem

V digitálním světě se často stává, že zatímco u hlavního vchodu se hlídá s vlčákem, zadní dveře zůstávají otevřeny dokořán. Snadnost a beztrestnost přístupu přes otevřené dveře může být však jen zdánlivá a sloužit k tomu, aby se útočník nalákal – a sedl „na lep".

ROBERT PETRUNIČ

K poznání chování útočníků slouží především tzv. honeypoty. V rámci projektu CS Danube (Kybernetická bezpečnost v Podunají) v rámci programu Start se CZ.NIC spolu s chorvatským partnerem zaměřil na pasivní obranu ve formě honeytokenů (česky nejspíše „nastražený údaj") a jejich integrace do webových aplikací.

Zatímco konvenční metody se snaží systém opevnit a bránit, honeytoken jde útočníkům naproti. Přestože honeytoken původem není doménou informatického světa, výborně poslouží. Upravená webová stránka totiž může samočinně odhalit a izolovat útok nebo ho dokonce obrátit proti útočníkovi.

Rozmanitost útoků

Webmaster musí své stránky zabezpečit proti různým typům útoků. Tato opatření nejčastěji začínají skenováním zranitelností webové aplikace. Většinou automatický nástroj otestuje všechny veřejné parametry a potom zaútočí některým ověřeným způsobem (pro jejich seznam viz například projekt Owasp Top 10).

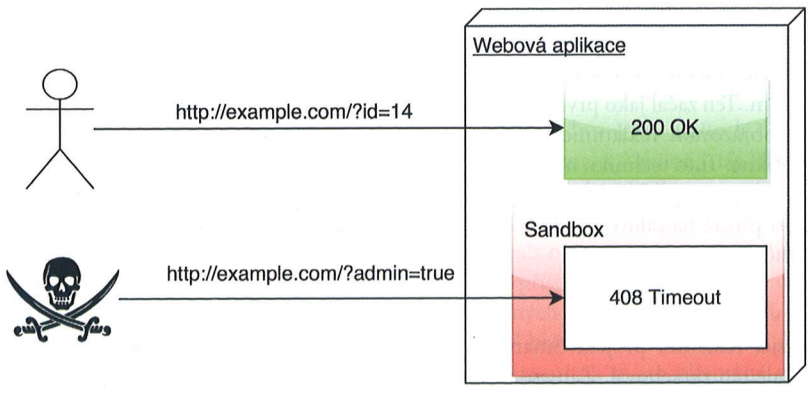
Server používající klasické způsoby obrany jako WAF (Web Application Firewall) nebo IDS (Intrusion Detection System) skenování poměrně snadno rozpozná a zabrání i známým atakům. Útokům dosud neznámým je ale vydán tzv. na milost.

Je téměř nemožné se připravit na všechny hrozby tradičními metodami a nástroji. Ale aby se útočníkům zvedla laťka alespoň o píď, lze napsat aplikaci, která se bude nějakým způsobem sama bránit.

Existuje mnoho nástrojů, které testují zranitelnost webové aplikace. Aplikaci nejprve proleze nějaký vyhledávací bot a shromáždí všechny údaje, které může. Pak se zkouší SQL injekce, XSS a známé exploity.

[Když honeytoken na jedné úrovni selže, další honeytoken pomůže odhalit selhání prvního a zalarmovat administrátory, aby následně vyšetřili systém.]

Honeytoken aktivuje sandbox



Některá řešení se pokoušejí odhalit i neodkazované složky a soubory, o kterých není nikde na webu zmínka – naslepo se připojují k souborům jako admin.php, a hledají tak vstupní body do neveřejné části aplikace, na něž pak zkusí zaútočit.

Další možností, jak zabránit tomuto střílení naslepo, je nastavit serveru HTTP status 200 OK pro každý požadavek, ať už směřuje k existujícímu zdroji nebo ne. Tato technika sken znatelně zpomalí. Také lze server nastavit tak, aby vracel fiktivní webové stránky, tedy neexistující vstupní body.

Co je honeytoken?

Ve většině případů si serverová část webové aplikace vyměňuje data s uživatelskou částí pomocí parametrů v URL, a to je také nejvhodnější místo, kam honeytoken umístit – jako jeden z parametrů s konstantní hodnotou (canary value).

Jeho použití je prosté. Systém hlídá, zda je hodnota neměnná, přičemž každá změna znamená, že si někdo zkouší s aplikací hrát.

Honeytoken může být v podstatě cokoli. Nejen parametr v URL, ale i údaj v databázi nebo textový soubor. Podle aktuálních potřeb se může koncept honeytokenu implementovat na každou úroveň aplikace, kde pomůže odhalit skenování zranitelností, spustit nějakou proceduru včasného varování a zabránit kompromitování systému.

Když honeytoken na jedné úrovni selže, další honeytoken pomůže odhalit selhání prvního a zalarmovat administrátory, aby následně vyšetřili systém.

Samotná implementace honeytokenů však ještě nepřinese detaily útoku na zlatém podnose. K tomu je zapotřebí zapojit software typu SIEM (Security Information and Event Management) nebo libovolný centrální logovací systém, který dokáže uspořádat logovací záznamy aplikace, webového serveru, operačního systému, databáze apod. do vztahů a názorných grafů, z nichž lze útok a jeho účinky vystopovat.

Hodí se ale vizualizovat i další informace – o geolokaci, routingu, množství generovaného přenosu apod.

Konkrétní příklad

Jak je možné takový honeytoken implementovat? Jako příklad si lze vzít webovou aplikaci, která na příslušné URL vrací data uživatele s identifikátorem 1:
https://example.com/Site.aspx?ID=1

Jedním z nejsnadnějších způsobů implementace honeytokenu je jeho umístění do parametrů.

Vývojář použije nějaké lákavé jméno, které útočníka navnadí, ovšem ohlídá, že hodnota parametru zůstane neměněná.

Nyní se přidá druhý parametr, který se bude tvářit, že má co do činění s administrátorskými právy: *https://example.com/Site.aspx?ID=1&Admin=false.* Útočník možná zatouží změnit hodnotu parametru na true, tím ale jen spustí samoobranný mechanismus aplikace.

Honeytokenem se může stát i údaj v databázi. Pokud k tomuto údaji někdo přistoupí, aplikace dostane zprávu, že byla napadena, a zvolí způsob obrany: zda útočníkovu relaci odpojí, a zabrání tím dalšímu úniku dat, nebo jestli začne sama sbírat stopy, aby útočníka dokázala identifikovat.

Podobně jako honeytoken lze využít i fyzický soubor, který monitoruje svůj přístup a vyvolá poplach. Přitom se nabízí využít možnosti konvenčního souboru robots.txt v kořenovém adresáři webu, jenž informuje boty, které stránky mohou navštívit a které nesmějí.

Jednoduše se botům předhodí, že nesmějí navštěvovat instalované honeytokeny. A když neuposlechnou – jsou to ti zlí.

Metody obrany

Co dělat, když se odhalí probíhající útok? Buď se může útočníkova relace ukončit, nebo ji tajně přesunout do nějakého zabezpečeného režimu a v něm při následné analýze nasbírat zajímavá data o způsobech útoku.

Případně se také může automaticky změnit nastavení firewallu webového serveru. Tady jsou některé možnosti:

Ukončení relace

Tento přístup má svá pro i proti. Výhodou je, že se session bude muset znovu vytvořit, pokud chce útočník ve své činnosti pokračovat. Ten si ovšem snadno dá dvě a dvě dohromady, parametry s honeytokenem odhalí a následně ignoruje.

Status 200 OK pro každý požadavek

Dotaz na libovolné URL vrátí stránku s HTTP statusem 200 OK, který značí, že stránka existuje. Přestože je metoda celkem účinná, lze ji také snadno odhalit, pokud fiktivní stránka obsahuje stále týž obsah.

Poněkud se vylepší, když se obsah pokaždé upraví, ale útočník nebo i stroj mohou rozpoznat, že ho obránci zkoušejí napálit. Pak je ovšem otázka, nakolik dobře rozpozná vzorec obsahu a odliší fiktivní stránky od skutečných.

Status 200 OK pro náhodné požadavky klienta

Metoda je téměř shodná s předchozí, ovšem efektivnější, neboť když se vracejí fiktivní stránky jen někdy, je mnohem těžší odhalit, že server podvrhuje odpovědi.

Zvláště pokud se vyrobí nějaký slovník, na základě kterého bude server vracet status 200 pro stránky s lákavým názvem jako *example.com/AdminPage.aspx* raději než pro nepravděpodobné stránky jako *example.com/Tot%C3%A1ln%C3%AD_bl%C3%A1bol_xyz.aspx*.

Izolace relace, sandbox

Velmi efektivní je uzavřít útočníka do sandboxu. Při detekci útoku aplikace relaci zcela zaizoluje, zablokuje svou skutečnou funkčnost a útočníkovi vrátí jen předpřipravené (nebo dynamicky generované) falešné stránky.

Problémem zde je ale vysoká náročnost implementace komplexního podvrhu.

Zpomalení odezvy

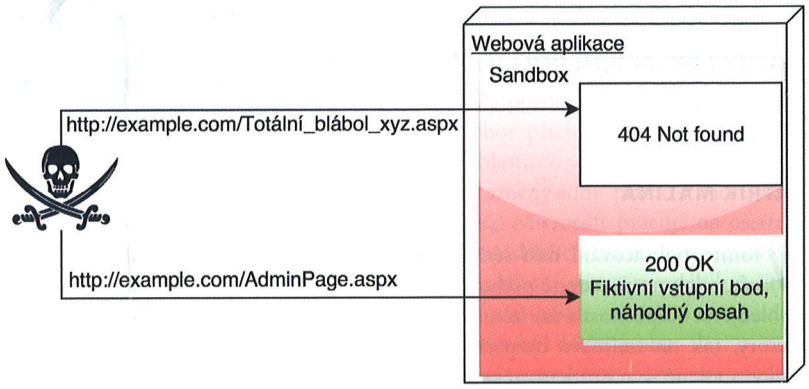
Možná úplně nejjednodušší obranou je implementovat sabotážní mechanismus, který útočníkovu činnost zpomalí k nepoužití. Webmaster na druhé straně získá čas a prostředky k protiakci.

Status 408 – Request Timeout

Útok navíc citelně poškodí, když se některé požadavky útočníka nezodpovědí záměrně vůbec nebo se mu předhodí falešné tvrzení, že jeho požadavek vypršel. Tento způsob je vhodné kombinovat se zpomalováním odezvy.

Útočníkovi se tak ztíží skenování, útok nebo zkrátka cokoli, co dělá.

Honeytoken – chování v sandboxu



[Velmi efektivní je uzavřít útočníka do sandboxu.]

Zpětný úder

Útočníci často skrývají svou skutečnou IP adresu za jednou nebo několika proxy. Aplikace může být naprogramovaná tak, že v případě útoku použije XSS zranitelnost, která při úspěchu vyzradí klientovu skutečnou IP adresu.

Stejným způsobem lze útočníkovi zcizit i cookies, e-mailový účet apod.

Síla je v kombinované obraně

Zmíněné metody popisují možnosti aktivní prevence proti automatickým nástrojům a/nebo uživatelům – útočníkům.

Zdálo by se, že honeytokeny v databázi a v souborovém systému nejsou třeba, protože každý útok se zastaví už na aplikační vrstvě.

Ovšem je nutné předjímat, že aplikační vrstva stále může mít zranitelnost – ba dokonce v obraně samé. Vyplatí se tedy implementovat co nejvíce nezávislých metod obrany, které doplní klasické nástroje – firewall a IDS totiž nemohou zabránit modifikaci parametrů a webový firewall (WAF) lze obejít.

Honeytokeny však zůstávají skryté hluboko v aplikační logice. Přinášejí administrátorům, vývojářům nebo bezpečnostním pracovníkům další výhodu a šanci, jak získat kompletní vhled do samotného útoku, jeho metod a technik a nasadit aktivní protiopatření automaticky přidaná do aplikace v momentě výskytu útoku.

A když si pozdě v noci jdete pro svačinu, je skvělé tušit, že si zloděj v kuchyni vaří kávu. ■

Autor pracuje jako bezpečnostní analytik sdružení CZ.NIC, které provozuje Národní bezpečnostní tým CSIRT.CZ.