Analýza postranních kanálů (kryptoanalýza hardwarových zařízení)

Jakub Breier, Juraj Ďaďo, Juraj Hreško, Matej Klement,
Jiří Kusák, Jan Michelfeit, Martin Moráček, Michal Vančo

21. května 2010

Úvod

Útoky postranními kanály jsou útoky, které jsou založeny na informaci o postranním kanálu. Informace o postranních kanálech jsou informace, které je možné získat ze šifrovacího zařízení, nejedná se o čistý text k zašifrování ani o ciphertext výsledného šifrovací procesu.

V minulosti bylo šifrovací zařízení vnímáno jako celek, které přijímá čistý text na vstupu a produkuje ciphertext na výstupu a naopak. Útoky byly tedy prováděny buď na základě ciphertextu (např. ciphertext-only útoky), nebo na základě obou (např. známé plaintext útoky), nebo na schopnosti definice jak má být čistý text šifrován (tzv. vybrané plaintext útoky). Dnes je známo, že šifrovací zařízení mají i jiné vstupy a výstupy, které nepracují s čistým textem nebo ciphertextem. Šifrovací zařízení produkují časovou informaci (informace o době trvání operace), která je snadno měřitelná, záření různých druhů, statistiky spotřeby energie (které lze také snadno měřit) a další. Šifrovací zařízení mají často také další neúmyslné vstupy, jako je například napětí, které může být změněno, aby způsobilo předvídatelné výstupy. Útoky postranním kanálem využívají některé nebo všechny tyto informace spolu s dalšími známými kryptoanalytickými technikami k získaní klíče, který zařízení používá.

Klasifikace útoků na hardwarové zařízení

Útoky na kryptografické zařízení se mohou lišit z hlediska ceny, času, potřebného vybavení a znalostí nutných k jejich realizaci. Je několik způsobů, jak kategorizovat tyto útoky, nejčastěji na základě dvou kritérií. Prvním kritériem je rozdělení útoků na aktivní a pasivní.

- Pasivní útoky – u těchto typů útoku pracuje kryptografické zařízení z převážné části nebo zcela standardním způsobem v rámci své specifikace. Tajný klíč je získaný pomocí sledování fyzických vlastností zařízení, jakými může být například čas výpočtu nebo spotřeba elektrického proudu.
- Aktivní útoky– v těchto případech je šifrovací zařízení, jeho vstupy nebo jeho prostředí manipulováno za účelem nestandardního chování. Takové chování je pak možné zneužít k získání tajného klíče.

Dalším kritériem rozdělení typu útoků je rozhraní, přes které útočník přistupuje k zařízení. Kryptografické zařízení mají obvykle několik logických a fyzických rozhraní, přičemž k některým se dá přistupovat jednoduše a k některým jen pomocí speciálního vybavení. Rozlišujeme invazivní, semi-invazivní a neinvazivní útoky, přičemž každý z těchto útoků může být aktivní nebo pasivní.

- Invazivní útoky – tento typ útoku je nejsilnějším typem, který může být vykonán na kryptografickém zařízení. Je u něj možné se zařízením udělat cokoliv pro to, aby byl získán tajný klíč pro upisování. Invazivní útoky typicky začínají obnažením části zařízení, aby bylo možné přistupovat k jeho jednotlivým komponentám. Při pasivním typu útoku se pouze sleduje datový signál z komponent, při aktivním se tento signál i uměle mění. Tento typ útoků je velmi mocný, avšak typicky vyžaduje drahé zařízení na své uskutečnění.
- Semi-invazivní útoky – i při tomto typu útoku je šifrovací zařízení obnažené, avšak není proveden žádný elektrický kontakt s povrchem čipu. Cílem pasivních typů útoku je typicky přečtení obsahu paměťových buněk bez použití klasických čtecích obvodů. Při aktivním typu semi-invazivních útoků je cílem vyvolat chyby v zařízení, což může být provedeno pomocí rentgenových paprsků, elektromagnetických polí nebo světelných signálů. Tyto typy útoku sice nevyžadují až tak drahé zařízení jako invazivní útoky, ale úsilí nezbytné pro jejich uskutečnění je stále poměrně vysoké.
- Neinvazivní útoky – u těchto typů útoku není napadeno kryptografické zařízení jako takové, jsou pouze využita přístupná rozhraní. Zařízení po útoku zůstává v nezměněném stavu, proto není zanechaný žádný důkaz o útoku. Většinu těchto útoků je možné vykonávat s relativně levným vybavením. Představují tedy největší hrozbu vůči bezpečnosti kryptografických zařízení.

  Pasivní neinvazivní útoky se nazývají útoky analýzou postranních kanálu. Tři nejdůležitější typy těchto útoků jsou časové útoky, elektromagnetické útoky a útoky odběrovou analýzou. Základním konceptem těchto útoků je získat tajný klíč pomocí měření času výpočtu, spotřeby elektrické energie nebo měření elektromagnetického bodového pole. Při aktivních útocích je účelem vyvolání chyby v zařízení, aniž by bylo nutné obnažit jeho vnitřní komponenty.

Nejčastější typy útoků

Časové útoky

Časové útoky jsou založeny na měření času výpočtu operací. Tato měření mohou vést k informaci o tajném klíči. Například pečlivým měřením času potřebného k provedení operace s privátním klíčem může útočník získat Diffie-Hellmanovy exponenty, faktor RSA klíče a nebo prolomit jiné kryptografické systémy. Je-li zařízení zranitelné, je útok výpočetně jednoduchý a často vyžaduje pouze známý kryptogram. Kryptosystémy se často mírně liší v množství času potřebném na zpracování různých vstupů. Mezi důvody těchto odlišností patří výkonové optimalizace k vyhnutí nepotřebným operacím, větvení a podmíněné příkazy, zásahy do RAM cache, procesorové instrukce (například násobení a dělení), které běží v různém čase, a celá řada jiných příčin. Výkonnostní charakteristiky obvykle závisí jak na šifrovací klíč tak i na vstupu dat (např. prostý text nebo kryptogram). Intuice by mohla naznačit, že neúmyslné časování odhalí jen malé množství informací z kryptografického systému. Nicméně existují útoky využívající měření času operací k získání tajného klíče z ohroženého systému.

SPA

Simple Power Analysis (Jednoduchá analýza spotřeby) je obecně založena na hledání vizuální reprezentace spotřeby energie na jednotku v průběhu šifrovací operace. Jednoduchá analýza spotřeby je technika, která zahrnuje přímou interpretaci měření spotřeby elektrické energie v průběhu kryptografické operace. SPA může přinést informace o operacích zařízení stejně jako o tajném klíči.

DPA

Differential Power Analysis (Diferenciální analýza spotřeby). Útoky touto technikou jsou hůře zabranitelné. Neskládají se jen z vizuální, ale také ze statistické analýzy, také obsahují statistické opravy chyb metod pro získání informací o klíči. DPA se obvykle skládá ze dvou fází – sběr dat a analýza dat, která používá statistické funkce pro filtrování šumu a pro získání dalších informací o procesech, které zařízení provádí.

Vedle vysoce spotřebních změn způsobených posloupností instrukcí, jsou zde také působení související s daty, se kterými se pracuje. Tyto varianty jsou obvykle menší a někdy jsou zastíněny chybami měření a šumem. V takových případech je možné prolomit systém pomocí statistických funkcí, které jsou šité na míru cílového algoritmu. Vzhledem k tomu, že DPA automaticky vyhledává korelované oblasti ve spotřebě energie zařízení, mohou být útoky automatizovány i s malými či žádnými informacemi o implementaci cílového zařízení.

Časové útoky

Příklad – Chinese Remainder Theorem

Čínský teorém o zbytku. Modulární snížení kroků obvykle způsobí nejvíce časových změn při modulárním násobení operací. Čínská věta o zbytcích (CRT) je často používána pro optimalizaci operací RSA soukromého klíče. S CRT $(y \bmod p)$ a $(y \bmod q)$ jsou počítány jako první, kde y je zpráva. Toto prvotní modulární snížení kroků může být zranitelné na načasování útoků. Nejjednodušší útok je výběr hodnot y, která jsou v blízkosti p nebo q, pak pomocí měření času operací zjistit, zda je získaná hodnota je větší nebo menší než je aktuální hodnota p nebo q. Jestliže y je menší než p, výpočet $y \bmod p$ nemá žádný vliv, zatímco pokud y je větší než p, bude nutné odečíst od p y alespoň jednou. Určité časové vlastnosti závisí na implementaci.

Příklad – Montgomery Multiplication

Ustálíme následující označení: zvažme výpočet $m^k \bmod n$, kde tajný exponent k má binární notaci $(k_{\omega-1}, k_{\omega-2}, \cdots, k_0)$, a $k_{\omega-1}$ označuje nejvýznamnější bit. Algoritmus obsahuje mocniny a násobení, obě operace se provádí pomocí Montgomeryho algoritmu.

V naivní implementaci je modulární násobení časově náročná operace. Montgomery navrhl způsob, jak urychlit tyto operace převedením do modulu, který je vhodnější pro vnitřní strukturu zařízení. Pro zjednodušení zde nebudeme popisovat Montgomeryho algoritmus v detailech. Postačí nám vědět, že pro libovolně zvolený modulus je čas pro Montgomeryho násobení konstantní nezávisle na faktorech, s výjimkou případu, kdy je mezivýsledek větší než modulus. Potom musí být provedeno další odčítání (tzv. redukce).

Nejběžnější způsob jak využít těchto znalostí je zamířit náš útoku na krok násobení při mocnění a násobení. Myšlenka je následující:

Začneme napadením $k_{\omega-2}$ druhého bitu tajného klíče. Provádíme mocniny a násobení algoritmu krok za krokem a vidíme, že pokud je bit roven 1, pak hodnota $m \cdot m^2$ bude muset být vypočítána v průběhu mocniny a násobení.

Nyní, pro některé zprávy m (ty, pro které bude mezivýsledek násobení větší než modulus) musí být provedeno další snížení v průběhu tohoto násobení, zatímco pro jiné zprávy není krok snížení potřebný. Takže jsme schopni rozdělit náš soubor vzorků do dvou podskupin podle toho, jestli výpočet $m \cdot m^2$ způsobí redukci, nebo ne. Pokud je hodnota $k_{\omega-2}$ opravdu 1, pak můžeme očekávat, že výpočetní doba pro zprávy z první skupiny (způsobující redukci) bude mírně vyšší než odpovídající doba pro zprávy z druhé skupiny.

Avšak pokud skutečná hodnota $k_{\omega-2}$ je 0, pak operace $m \cdot m^2$ nebude provedena. V tomto případě naše dělící kritérium bude nesmyslné: není žádný důvod, proč by m způsobující redukci pro operaci $m \cdot m^2$ také způsobovalo redukci pro operaci $m^2 \cdot m^2$ nebo pro jakoukoliv jinou operaci. Proto by rozdělení do dvou podskupin mělo vypadat náhodně a neměli bychom pozorovat

žádný významný rozdíl ve výpočetní době.

Opatření proti časovým útokům

Přidání zpoždění

Nejjednodušší způsob, jak zabránit časovým útokům, je zařídit, aby všechny operace trvaly přesně stejnou dobu. Toto bývá naneštěstí často obtížné. Pokud je použit časovač, aby zpozdil navrácení výsledků na specifikovaný čas, je pořád možno detekovat změny faktorů jako např. odezvy systému nebo spotřeby energie. Také implementace s fixní časovou hodnotou bývají pomalé; mnoho optimalizací nemůže být použito, jelikož všechny operace musí trvat stejně dlouho jako ta nejpomalejší z nich. Pokud se přidají náhodná zpoždění, mohou na ně útočníci reagovat sběrem více vzorků (přestože počet potřebných kryptogramů bude vyšší). Jejich požadovaný počet se zvyšuje zhruba jako druhá mocnina časového šumu. Náhodné zpoždění tedy může útok trochu ztížit, ale neznemožní jej.

Vyrovnání časové náročnosti násobení a mocnění

Čas, který zabere zařízení vykonání operací násobení a mocnění, by měl být nastaven na podobnou hodnotu. Díky této vlastnosti nebude útočník schopen rozpoznat kdy a kolik operací násobení a mocnění bylo provedeno. Vyrovnání může být dosaženo prováděním vždy obou operací nezávisle na tom, která z nich je právě požadována. V každé fázi, kdy je jedna z operací vyžadována, by měly být provedené obě a výsledek té nepožadované operace by byl potichu ignorován. Tato technika zabraňuje časovým útokům proti operacím mocnění, které jsou prováděny jako část asymetrických šifrovacích operací a které jsou terčem nejběžnějších útoků.

SPA

Útočník přímo pozoruje spotřebu energie systému. Hodnota spotřeby se mění v závislosti na právě vykonávané instrukci mikroprocesoru. Velké operace jako např. DES rundy, RSA operace atd. mohou být identifikovány, protože operace prováděné mikroprocesorem se mohou výrazně lišit v jednotlivých částech těchto velkých operací. SPA analýza může být použita např. k prolomení RSA implementací odhalením rozdílů mezi násobícími a mocnícími operacemi. Podobně tak mnoho DES implementací má viditelné rozdíly mezi permutacemi a posuny a tím pádem může být prolomena za pomocí SPA.

Protože SPA může odhalit pořadí provedených instrukcí, může být použita k prolomení kryptografických implementací, ve kterých je větev výpočtu závislá na zpracovávaných datech. Např. DES key schedules, DES permutace, porovnání, násobičky, mocničky.

- **DES key schedule:** výpočet DES key schedule zahrnuje rotaci 28bit registrů klíče. Podmíněné větvení je běžně používáno ke kontrole přetečení na jedné straně (tak, že se ten 1 bit doplní z druhé strany). Výsledná spotřeba energie pro bity 0 a 1 se bude lišit, pokud se pro tyto bity budou lišit i odpovídající větve výpočtu.
- **DES permutace:** DES implementace provádějí řadu bitových permutací. Podmíněné větvení v software nebo mikrokódu může způsobit značné rozdíly ve spotřebě energie pro bity 0 a 1.
- **Porovnání:** operace porovnání řetězců nebo bloků paměti běžně provádějí podmíněné větvení v případě nalezení rozdílného prvku. Toto větvení má za následek výrazné SPA (a někdy též časové) charakteristiky.

- **Násobičky:** obvody pro modulární násobení mají tendenci k úniku informací o datech, které zpracovávají. Únikové funkce závisí na návrhu násobičky, ale často jsou v těsném vztahu k hodnotám operandů Hammingovým vahám.
- **Mocničky:** jednoduchá modulární mocninná funkce prochází exponent a provádí v každé iteraci druhou mocninu spolu s dodatečným násobení pro každý bit exponentu roven 1. Exponent může být kompromitován, pokud mocnění a násobení má odlišnou spotřebu energie nebo je od sebe odděleno rozdílným kódem. Modulární mocninné funkce operující na dvou a více bitech současně mohou mít komplexnější funkce úniku informací.

DPA

Útoky diferenciální odběrovou analýzou jsou nejpopulárnějším typem odběrové analýzy. Je to hlavně z důvodu, že nevyžadují detailní znalost napadeného zařízení a dokáží odhalit tajný klíč i v případech, že zachycené vzorky obsahují vysoký podíl šumu.

Oproti útokům jednoduchou odběrovou analýzou vyžadují velký počet vzorků a je tedy často nevyhnutelné mít na určitý čas fyzickou kontrolu nad zařízením. Při těchto útocích je obvykle nutná pouze znalost šifrovacího algoritmu. Zaznamenané vzorky jsou analyzované odlišným způsobem jak při útocích jednoduchou odběrovou analýzou, kde se spotřebovaná energie analyzuje hlavně vzhledem k časové ose a útočník se snaží najít shodu v grafech jednoho vzorku. Při útocích diferenciální odběrovou analýzou není tvar vzorku vzhledem k časové ose až tak důležitý. Analyzuje se, jak spotřebovaná energie ve stanovených časových úsecích závisí na zpracovávaných datech, takže tyto útoky se zabývají výlučně datovou závislostí získaných vzorků. Používají se statistické metody, které pomáhají efektivně využít korelaci mezi daty a pravidelnými úniky informací z postranních kanálů. Útočník v tomto případě využívá hypotetický model zařízení, který pomáhá predikovat výstup z postranního kanálu zařízení. Kvalita tohoto modelu závisí na možnostech a znalostech útočníka. Výstupy mohou být případě hodnoty popisující jeden typ informace pro několik časových úseků, či hodnoty predikující výstup z více postranních kanálů. V případě, že je na útok použita jedna výstupní hodnota, hovoříme o útocích prvního řádu. V případě, že jsou na útok použité dvě a více hodnot ze stejného kanálu, mluvíme o útocích druhého, respektive vyšších řádů. Na rozdíl od jednoduché odběrové analýzy existuje při diferenciální odběrové analýze všeobecná strategie útoku, která je používaná při všech útocích tohoto typu. Tato strategie se skládá z následujících pěti kroků:

1. **Zvolení výsledku vykonávaného algoritmu**
   Prvním krokem je zvolení mezivýsledku vykonávaného algoritmu na napadeném zařízení, přičemž tento mezivýsledek je funkce $f(d, k)$, kde d je známá nekonstantní datová hodnota a k je část klíče. Mezivýsledek splňující tuto podmínku může být použitý na zjištění hodnoty k. Ve většině případů je d buď to otevřený, nebo zašifrovaný text.

2. **Měření spotřebované energie**
   Druhý krok se skládá z měření spotřebované energie kryptografickým zařízením během šifrování, nebo dešifrování D odlišných datových bloků. Během těchto operací musí útočník vidět datové hodnoty d, které se podílejí na výpočtu mezivýsledků zvolených v prvním kroku. Tyto hodnoty zapisujeme jako vektor $d = (d_1, \ldots, d_D)'$, kde d_i je datová hodnota v i-tém kroku šifrování, resp. dešifrování. Při každém běhu zaznamenává útočník odběrové vzorky. Ty vzorky, které odpovídají datovému bloku d_i, označujeme $t'_i = (t_{i,1}, \ldots, t_{i,T})$, přičemž T je délka vzorku. Útočník změří vzorky pro každý datový blok a výsledkem je matice T velikosti $D \times T$. Pro útoky diferenciální odběrovou analýzou je důležité, aby byly odběrové vzorky správně zarovnané. To znamená, že chceme, aby ve sloupci t_j matice T byly hodnoty získané tou samou operací.

3. **Výpočet hypotetických mezivýsledků**
   V tomto kroku se počítají hypotetické mezivýsledky pro každou možnou volbu k. Tyto potenciální hodnoty k zapisujeme jako vektor $k = (k_1, \ldots, k_K)$, kde K je počet všech možných

hodnot k. Tomuto vektoru říkáme vektor klíčových hypotéz. Pomocí datového vektoru d a klíčových hypotéz může útočník snadno vypočítat hypotetické mezivýsledky $f(d, k)$ pro všechny běhy D a pro všechny klíčové hypotézy K. Výsledkem těchto výpočtů je matice V velikosti $D \times K$. Sloupec j v matici V obsahuje mezivýsledky vypočítané pomocí klíčové hypotézy k_j. Vzhledem k tomu, že klíč použitý v kryptografickém zařízení je prvkem k, obsahuje jeden sloupec matice reálné hodnoty, které zařízení vypočítalo. Klíč z dané množiny odpovídající tomuto sloupci označíme k_{ck}. Když zjistíme, který sloupec z V byl zpracovaný v zařízení, dostaneme i k_{ck}, což je cílem útoku.

4. **Výpočet spotřebované energie z hypotetických mezivýsledků**
   V tomto kroku se z matice hypotetických mezivýsledků V vytvoří matice H, která obsahuje vypočítané hypotetické hodnoty spotřebované energie. Pro tento účel je nutné použít simulační techniky, na jejichž základě dokážeme vytvořit model zařízení, který umožní vypočítat s určitou přesností spotřebovanou energii. Při těchto útocích není důležité určit absolutní spotřebovanou energii, ale stačí pouze znát relativní rozdíly mezi odběrovými hodnotami. Běžně se používají dva modely, kteří jsou vhodné na diferenciální odběrovou analýzu: model využívající Hammingovu vzdálenost a model využívající Hammingovu váhu.

5. **Porovnání hypotetických hodnot se zachycenými vzorky**
   V posledním kroku je porovnáván každý sloupec h_i matice H se sloupcem t_j matice T. To znamená, že útočník porovná hypotetické hodnoty spotřebované energie pro každou klíčovou hypotézu se zachycenými vzorky na každé pozici. Výsledkem tohoto porovnání je matice R velikosti $K \times T$, přičemž každý prvek $r_{i,j}$ obsahuje výsledek porovnání mezi sloupci h_i a t_j. Existuje několik algoritmů, které se používají na toto porovnání. Všechny mají tu vlastnost, že hodnota $r_{i,j}$ roste s kvalitou shody h_i a t_j. Zachycené vzorky odpovídají spotřebované energii zařízení během vykonávání kryptografického algoritmu pro různé vstupy. Mezivýsledky zvolené v prvním kroku jsou součástí algoritmu. To znamená, že zařízení musí vypočítat mezivýsledky v_{ck} při různých běžích algoritmu a zachycené vzorky musí na nějaké pozici záviset na těchto mezivýsledcích. Tuto pozici označujeme jako ct, takže sloupec t_{ct} obsahuje hodnoty spotřebované energie závisející na mezivýsledcích v_{ct}. Hypotetické hodnoty spotřebované energie h_{ck} byly nasimulované na základě hodnot v_{ck}, takže sloupce h_{ck} a t_{ct} spolu silně souvisí. Nejvyšší hodnota v R bude potom právě výsledkem porovnání těchto dvou sloupců a můžeme ji označit $r_{ck,ct}$. Útočník může tedy snadno odhalit index klíče ck a časový moment ct hledáním nejvyšší hodnoty v matici R. Indexy této hodnoty jsou výsledkem útoku diferenciální odběrovou analýzou. V případě, že hodnoty v matici R jsou zhruba stejné, znamená to, že útočník nenaměřil dostatečný počet vzorků na vyhodnocení vztahu mezi sloupci matic H a T.

Opatření proti analýze spotřeby

Vyhýbání se podmíněnému větvení a skrytým meziproduktům

Vyhýbání se procedurám, které využívají skrytých meziproduktů nebo klíče k podmíněnému větvení, zamaskuje mnoho SPA charakteristik. Softwarové implementace kritického kódu by neměly obsahovat příkazy způsobující větvení programu a obdobně ani příkazy pro podmíněné vykonávání kódu (jako jsou např. IF klauzule). Výpočty by měly být prováděné pomocí funkcí využívajících elementárních operací (jako je AND, OR a XOR) a nepoužívajících větvení, ani podmíněného vykonávání částí kódu. Tato opatření mohou velmi ztížit odhadování hodnot vstupu a klíče pomocí měření času nebo spotřeby energie. Když se vždy provedou všechny řádky kódu bez ohledu na vstup nebo bity klíče, čas a spotřeba energie na datech nezávisí a tedy ani neodhaluje žádné jejich vlastnosti. Tato opatření zabraňují všem druhům časových útoků na asymetrické šifry a stejně tak i většině útoků na základě spotřeby energie.

Vyrovnávání spotřeby energie

Techniky na vyrovnávání spotřeby energie by měly být aplikované, kde je to jen možné. Nadbytečné operace (z hlediska algoritmu) by měly být prováděny nad dodatečnými falešnými registry a branami tak, aby měla spotřeba energie pokud možno konstantní hodnotu. Kdykoliv je operace prováděna v hardware, komplementární operace by měla být prováděna na falešném elementu, aby se zajistila vybalancovaná celková spotřeba vzhledem k nějaké vyšší hodnotě. Takové techniky, při kterých je spotřeba energie (z venkovního pohledu) konstantní a nezávislá na vstupu a bitech klíče, zabraňují všem možným způsobům útoků podle spotřeby energie jako jsou i SPA a DPA.

Redukce velikosti signálu

Jeden způsob prevence před DPA útoky je redukce velikosti signálu např. používáním kódu s konstantní vykonávanou cestou, výběrem operací, které způsobují menší únik informací ve své spotřebě energie, vyrovnáváním Hammingových vah a přechodů mezi stavy nebo fyzickou ochranou zařízení. Naneštěstí podobná redukce signálu nemůže v obecném případě redukovat signál na nulovou hodnotu, takže útočník s nekonečným počtem vzorků bude i nadále schopen provést DPA na signálu (byť velmi degradovaném).

Přidání šumu

Další přístup proti DPA zahrnuje přidání šumu do výsledných hodnot spotřeby energie. Podobně jako redukce velikosti signálu i přidávání šumu zvyšuje počet vzorků potřebných pro útok. Časování výpočtu (a pořadí instrukcí) může být také náhodně transformováno, aby se dosáhlo podobného efektu. Dalším možným řešením je přidat náhodné výpočty, které by opět přidaly dostatek šumu do spotřeby energie. Samotný šum však kromě zvýšení počtu vzorků žádný další vliv nemá. Každopádně pokud je toto navýšení dostatečně velké, aby znemožnilo realizaci útoku počtem potřebných snímků, je protiopatření účinné. Hlavním cílem je přidat dostatek náhodného šumu, aby byl útok zmařen, ale zároveň omezit režijní náklady na minimum.