1 Tučňáci z Madagaskaru

Daniel Skýpala

27. 2. 2022

1 Útok na ukázkový protokol

Nejdřiv se budu věnovat útokům, protože části navrženého protokolu se snaží mu zabránit a lze se na útoky odkazovat.

1.1 Replay attack

Jednom rychlé zmínění, abych se na tuhle chybu mohl odkazovat.

1.2 Neviděl jsem to už někde...

Můžeme si všimnout, že když pošleme ukázkovým protokolem dvakrát stejnou zprávu, zašifruje se stejně. Pokud si nepřítel bude zapamatovávat všechny přijaté zprávy a zároveň reakce základny na ně, je schopen odhadnout co základna udělá a tedy i její typ.

1.3 Na délce záleží.

Dále si můžeme všimnout, že zprávy, které budeme posílat jsou různě dlouhé. Např. *VYCKEJTE* má 8 znaků, zatímco *OPAKUJ* 6. Podle toho je možné určit typ posílané zprávy (a v případě *POSLETE MI* n počet řádů n).

1.4 Že by chyba v přenosu?

Útočník může zkoušet měnit zprávu. Sice většinou budou vycházet nesmylsly, ale zpráva *POSLETE MI* n je na toto obzvláště citlivá, protože pokud útočník bude měnit zprávu (prohazovat znaky zprávy za náhodné znaky) v oblasti, kde je uloženo n a bude mít štěstí, tak je schopen n změnit na jiné číslo.

Následující chyby ukázkový protokol má, ale zranitelností jde dosáhnout jednodušeji využitím výše zmíněných chyb. Sem je jenom dávám, abychom se jich vyvarovali při navrhování vlastního protokolu.

1.5 Double time

Potom co bude poslána zpráva, tak si ji uložíme a nepošleme ji dál. Poté počkáme, až ji výzkumná skupina pošle znovu a v tu chvíli pošleme obě zprávy naráz. Pokud se jedná třeba o zprávu *POSLETE MI n*, efektivně jsme poslali základně *POSLETE MI 2n*, čímž donutíme základnu udělat něco jiného.

1.6 Držení zajatců

Podobně jako při předchozím útoku zadržíme poslanou zprávu. Situace se ale změní a výzkumná skupina bude chtít poslat jinou zprávu. V takovém případě může nepřítel buď poslat první zprávu, obě zprávy, nebo obě zprávy v opačném pořadí. (Když výzkumná skupinka chce, aby základně došla jen druhá zpráva.)

A následující chyby ukázkový protokol nemá, ale chceme se jich vyvarovat při složitějším protokolu:

1.7 Neviděl jsem to už někde... (second edition)

Vzhledem k tomu, že zašifruj šifruje písmenko po písmenku, tak pokud je stejná část zprávy, ve které je obsah, tak má stejný obsah. (Např. Pokud při každé zprávě přidáme k obsahu náhodné znaky vlevo a vpravo a zašifrujeme, tak pro stejný obsah prostředek zprávy je stále stejný.) Toto můžeme využít stejně jako v původní verzi útoku.

1.8 Slepování

Pokud víme, jakou část tvoří ve zprávě tvoří samotný obsah zprávy (např. *VYCKEJTE*), můžeme vzít obsah první zprávy a vyměnit ho s obsahem druhé zprávy, s tímže zbytek necháme (pokud je ve zprávě např. pořadové číslo). Tím by se dala prolomit určitá ochrana pro pořadí. (Pokud např. posíláme pořadové číslo zprávy.)

1.9 Ten internet už zase nefunguje

A na závěr se nesmí stát, že pokud jedna zpráva nedojde, nebo je přerušena, protokol se zhroutí. (Např. Pokud by základna čekala na zprávu č. 2, ale výzkumná skupina by posílala protokoly s vyšším číslem.)

2 Návrh protokolu

Na začátku si tučňáci domluví strategii a vymění si hlavní šifrovací klíč *mainkey*. Samotná komunikace vypadá následovně: Když výzkumná skupinka chce poslat zprávu základně, zašifruje ji protokolem níže a pošle ji a bude ji opakovaně posílat tak dlouho, než základna nepotvrdí přijetí zprávy ve validním formátu.

Potvrzení zprávy probíhá tak, že pokud je zpráva validní, základna ji pošle zpátky výzkumné skupince ve stejném znění, jak ji dostala.

Odesíláná zpráva vypadá takto:

basemessage := [id][timestamp][message][padding][hash]

protectedmessage := [sifruj(key, basemessage)][key]

sendedmessage := sifruj(mainkey, protectedmessage)

Její části:

2.1 Message

Samotný obsah zprávy, který chceme poslat. (Např. *VYCKEJTE*) Délka 6 - 26 znaků.

2.2 Padding

Tečka + náhodně vygenerovaná zpráva, takové délky, aby spolu s message dávala 64 znaků. (Celkem délka paddingu: 64 - len(message), náhodné zprávy o 1 kratší). Zabraňuje útokům, které využívají různé délky zpráv a útokům, že stejná zpráva vypadá stejně. (1.2, 1.3)

Tečka na začátku paddingu je proto, abychom poznali, kde končí odesílaná zpráva a začíná padding.

2.3 Id

Pořadové číslo zprávy. Délka 13 znaků. Na začátku je nastaveno na 0, pokaždé když je zpráva potvrzena, zvýší si ho výzkumná skupina i základna o 1. Pokud dojde zpráva s takovým *id*, jaké očekáváme, tak může být validní, jinak je tato zpráva zastaralá, nebo falešná a budeme ji ignorovat. Slouží k obraně proti útkoům: 1.1, 1.5

2.4 Timestamp

Čas, do kdy je zpráva platná ve formátu dd/mm/yyyy hh:mm:ss. Délka 19 znaků. Slouží k tomu, že když posíláme určitou zprávu s daným id, ale situace se změní a chceme polat jinou zprávu než jsme posílali původně. (V takovémto případě nechceme, aby základně nedošla první zpráva nebo obě.) Tak můžeme počkat, než vyprší platnost první zprávy a začneme posílat novou druhou. (Útok 1.6)

Otázkou je, jak dlohou nastavit platnost zprávy. Když posíláme zprávu, platnost můžeme nastavit jako čas, mezi kterým byla poslední zpráva odeslána a potvrzena. (Pro případ první odesílané zprávy dáme rozumnou hodnotu podle odhadu připojení.) Pokud vidíme, že nám nechodí žádné potvrzení, tak platnost zvýšíme (např. vynásobíme 2×). Zároveň ale nechceme, aby platnost byla

zbytečně velká, (kvůli obraně proti útoku 1.6, kdy čekáme na vypršení platnosti), takže je dobré zvolit nějakou maximální hodnotu, nad kterou zvyšovat nebudeme (třeba 5 sekund soudě podle normálnímu připojení k internetu).

2.5 Hash

Otisk celé následující části: [id][timestamp][message][padding]. (32 znaků) Zabraňuje tomu, že když je něco ve zprávě změněno, zpráva se tváří jako validní. (Útoky 1.4 a 1.8)

2.6 Protectedmessage

Následně je vygenerován náhodný klíč, basemessage je jím zašifrován a ke zprávě je připojen na toto použitý klíč. (Ten má 32 znaků.) Toto zabraňuje útoku 1.7.

Pozor - náhodné klíče nemůžeme znovupoužít, protože by bylo vidět, že jsme tak učinili. (Konec zprávy je stejný.) Poté bychom mohli aplikovat útok 1.7, protože vnitřek je stejný. Musíme si tedy udržovat klíče, které jsme již použili a generovat takový, který je nepoužitý.

2.7 Sendedmessage

Protože protectedmessage jde rozšifrovat kýmkoliv (obsahuje klíč) a přečíst, je zapotřebí ji celou zašifrovat ještě jednou hlavním klíčem, který jsme si vyměnili na začátku.

Nyní k ostatním věcem okolo protokolu:

2.8 Jak zprávu vytvořit?

Vezmeme si současné id, do kdy má zpráva platit, obsah zprávy a padding příslušné délky. Toto celé pospojujeme dohromady a uděláme z toho otisk, který připojíme za to. Dále si vygenerujeme náhodný klíč a zprávu i s otiskem náhodným klíčem zašifrujeme. Za zašifrovanou zprávu připojíme náhodný klíč. Nakonec všechno zašifrujeme hlavním klíčem a pošleme.

2.9 Jak zprávu rozšifrovat?

To co jsme dostali dešifujeme hlavním klíčem, poté ze zprávy vezmeme náhodný klíč a zbytek dešifrujeme náhdoným klíčem. Nyní ověříme validitu:

- Id je rovno očekávanému.
- Zprávě nevypršela platnost (timestamp).
- Hash sedí.

A pokud je zpráva validní, tak přečteme její obsah a pošleme potvrzení výzkumné skupince (Zpráva v přesném znění, tak jak nám přišla). Následovně jak my (hned po odeslání potvrzení), tak výzkumná skupinka si zvýší id o 1 (hned po přijetí potvrzení).

3 Obrana jednotlivým útokům

Jen v rychlosti proletím, jakým způsobem se protokol brání zmíněným útokům:

3.1 Replay attack

Id už není aktuální (původní zpráva byla přijata), zpráva je tedy nevalidní.

3.2 Neviděl jsem to už někde...

Padding je jiný, a po přešifrování náhodným klíčem je zpráva úplně jiná.

3.3 Na délce záleží.

Díky paddingu mají všechny zprávy stejnou délku.

3.4 Že by chyba v přenosu?

- Buď je poškozena část s id, platností, obsahem a paddingem. V takovémto případě nesedí hash, který se z těchto věcí dělá.
- Nebo je poškozen hash. V takovémto případě hash ale také nesedí.
- Nebo je poškozen náhodný klíč. V tomto případě po rozšifrování base-message vychází náhodný šum, takže zpráva nemá šanci projít kontrolami.

3.5 Double time

Zprávy mají stejné id, jen jedna z nich bude validní.

3.6 Držení zajatců

Poté, co chceme změnit zprávu počkáme až původní zprávě vyprší platnost (plus rezerva). (V tu chvíli víme, že základna zprávu nepřijala.) Původně poslané zprávě již vypršela platnost, je tedy nevalidní a můžeme posílat pouze novou. (Celý tento postup lze opakovat, pokud bychom chtěli změnit obsah víckrát.)

3.7 Neviděl jsem to už někde... (second edition)

Celá zpráva je přešifrována náhodným klíčem, takže obsah vždy závisí na náhodném klíči. Tedy stejný obsah je po zašifrování různý pro různé náhodné klíče.

3.8 Slepování

Buď slepujeme v rámci basemessage - v tomto případě nesedí hash zprávy, zpráva je tedy nevalidní. Nebo se snažíme vyměnit náhodný klíč - v tu chvíli vychází náhodný šum.

3.9 Ten internet už zase nefunguje

Tomuto útkou zamezujeme tak, že zprávu posíláme pořád dokola, než dostaneme potvrzení o přijetí.

4 Věci na zmínění

4.1 Kapacita

Pokud pošleme přílišné množství zpráv, protokol se začne rozbíjet - nejdříve nám dojdou náhodné klíče, poté paddingy a nakonec idčka. Toto způsobí pád celého protokolu, protože začne být zranitelný vůči opakovacím útokům (1.1, 1.2, 1.7,...). Protože ale můžeme poslat 2^{32} zpráv, což je při jedná zprávé na sekundu více než 136 let, tak tohle není úplně relevantní. Pokud bychom tomu přesto chtěli tomuto zabránit, můžeme délky příslušných částí zdvojnásobit (což sice zase není na věčnost, ale co už, vydrží to 2^{32}-krát déle).

4.2 Komunikace ze základny

Základna může a nemusí komunikovat. Má ten luxus předpokládat, že její zprávy nebudou měněny, takže pokud chce, stačí použít jen odeslat zprávu ve stejném protokolu, jako popsáno zde, akorát bez potvrzení. (Místo kterého si základna i skupina zvýší id o 1 - základna hned po odeslání, skupina hned po přijetí. - Toto je možné udělat, protože zprávy nejsou rušeny.)