Projekt do PB173
Bezpečná videokonferenční architektura

Tučňáčí kolektiv

12. října 2013

Obsah

1 Motivace

V současných dnech jsme svědky vzrůstajícího napětí mezi uživateli internetu. Toto napětí má na svědomí americký občan Edward Snowden, bývalý analytik CIA toho času v exilu v Rusku, který zveřejnil jak USA odposlouchává uživatele internetu bez ohledu na státní příslušnost. Tyto informace rozpoutaly vlnu vyšetřování, které odhalilo, že vláda USA, konkrétně její tajná služba NSA[1], která dle platných zákonů má právo přístupu, k veškeré komunikaci zprostředkované americkými firmami. Její pravomoce sahají tak daleko, že má právo i na zabudování vlastních mechanizmů do šifrovacích metod používaných americkými firmami, což jí umožňuje dešifrovat jinak nepřístupná data. Tento stav evokuje značné nebezpečí a je potřeba jej řešit. Již jsme si zvykli, že USA má přístup k naší běžné komunikaci, pro kterou používáme Facebook, Google či Skype, že se může podívat téměř na kteroukoliv fotografii, kterou pořídíme svým smartphone, ale co udělat, když ani naše tajná komunikace s milenkou, milencem, oblíbeným nočním podnikem nebo místní muslimskou komunitou, je beztrestně čtena kterýmkoliv z tisíců zaměstnanců amerických tajných služeb?
To nás přivádí k hlavní myšlence tohoto dokumentu, návrh API pro programování bezpečné videokonferenční aplikace. Toto API bude rozvrženo pro programovací jazyk C++ v němž budou ve značné míře použity části syntaxe jazyka C. Důvodem je využití knihovny PolarSSL, která je napsána v jazyce C. Hlavním cílem této práce je rozvrhnout a implementovat videokonferenční službu s ohledem na bezpečnost dat. Proto je nezbytný robustní návrh a využití pokročilých kryptografických metod.

2 Použite technologie:

C/C++ s knihovnou PolarSSL, v nasem připade vetšinově C++, ale jelikož je PolarSSL psána v C, tak sekce kolem šifrovani nemohou používat mechanismy C++ (např. použití pole znaků místo objektu typu String). Dále planujeme použít knihovny pro praci se sockety/HTTPS, knihovnu pro vlákna a události. Pro zpracování obrazu a zvuku využijeme doporučené knihovny ze cvičení.

Konkrétneji ke kryptografii: pro zajištení důvěrnosti, integrity i soukromí využijeme šifrovacího schématu encrypt-then-mac v podobe otevřené implementace Galois/Counter Mode (GCM), která je dostupná v PolarSSL. Jako hashovaci funkci budeme používat SHA-512, symetrickou šifrovaci funkcí

[1]National Security Agency - vnitronárodní rozvětka analogie naší BIS

bude AES v již zmíněném módu GCM a pro ustanovení komunikace použijeme asymetrické kryptografie v podobě RSA.

3 Náš přístup k řešení:

V projektu vystupují tři komponenty, mezi kterými je ustanovena komunikace.

Certifikační autorita Certifikační autorita (CA) je komponentou, která zajišťuje ověřování a distribuci certifikátů veřejných klíčů potřebných pro navázání asymetricky šifrované komunikace mezi klienty. Certifikační autorita přijímá certifikáty od nově příchozích klientů, a na žádost vydává veřejné certifikáty klientů. Pro komunikaci s CA je zřízen jeden pár veřejného a soukromého klíče, kerý je používán CA. Veřejný klíč je pro klienta známý. Komunikace je ustavena klientem, který po CA požaduje ověření veřejného klíče jiného klienta. Na tento dotaz CA odpovídá dohledáním příslušného veřejného klíče ve své databázi a odesláním certifikátu tohoto klíče.

Server Server je komponentou zaštiťující ustanovení komunikace mezi klienty. Komunikace probíhá pomocí jedsnosměrně šifrované komunikace od klienta k serveru. Server přijímá od klienta požadavky na přihlášení do systému a odesílá ostatním připojeným klientům jeho dostupnost. Následně očekává požadavky na spojení. Nakonec po odpojení odešle všem klientům požadavek na smazání klienta. Server přijme požadavek na spojení od klienta ke klientu. Tento požadavek propaguje k cílovému klientovi společně s údaji pro navázání komunikace se zdrojovým klientem. V případě přijetí pozitivní odpovědi odešle komunikační údaje cílového klienta zdrojovému a označí oba klienty za nedostupné. Pokud některý z klientů ukončí komunikaci odešle tuto informaci serveru a ten označí oba klienty za dostupné.

Klient Komponenta klient zajiťuje rozhraní pro přímou komunikaci dvou klientů, navázání komunikace a získávání veřejných klíčůnvystupujících v komunikaci. Klient se přpojí k serveru a přijme seznam připojených klientů.

Klient požaduje připojení ke klientovi: Klient požaduje připojení ke klientovi: Iniciální klient (IK) si vyžádá od CA veřejný klíč cílového klienta (CK) a zašle serveru požadavek na spojení. Po kladné odpovědi serveru a přijetí komunikačních údajů o CK. Následně navazuje komunikaci s CK. Odešle

požadavek spoelečně s náhodně vygenerovaným klíčem a inicializačním vektorem pro symetrickou kryptografii zašifrovaný a podepsaný pomocí asymetrické kryptografie k CK. Pokud CK odpoví kladně, přijme a zpracuje symetrický klíč a inicializační vektor pro komunikaci od CK a IK je schopen pomocí veřejného klíče CK ověřit autenticitu. Nakonec navazuje a přijímá komunikaci k resp. od CK šifrovanou pomocí symetrické kryptografie.

Klient přijímá požadavek na komunikaci od serveru: CK přijme od serveru požadavek a informace o IK. Od CA si vyžádá certifikát veřejného klíče IK a odpoví na požadavek. Následně přijímá požadavek přímo od IK, které ověří pomocí jeho veřejného certifikátu. Přijme klíč pro komunikaci od IK, vygeneruje a odešle klíč pro komunikaci mezi CK a IK. Nakonec přijímá a otevírá přímé spojení mezi IK a CK.

4 Naše API:

Viz hlavičkové soubory na GitHubu včetně dokumentace v repozitáři po názvem PB173_tucnaci dostupný zde: `https://github.com/LukeMcNemee/PB173_tucnaci`