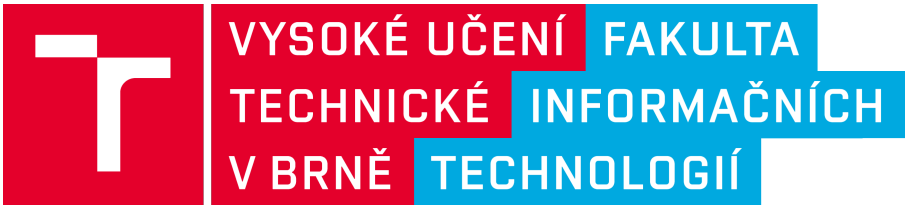

KRY Projekt 2

Vojtěch Fiala (**xfiala61**)

1 Šifrování a dešifrování

V projektu jsou využívány dva různé druhy šifrování – symetrické a asymetrické. Symetrické šifrování zajišťuje šifrování AES a asymetrické šifrování pak šifra RSA.

1.1 AES

Šifra AES[1] je symetrická šifra, tedy pro šifrování i dešifrování používá stejný klíč. Její fungování sestává ze 4 částí:

- Byte Sub – každý vstupní bajt je nahrazen jiným na základě substituční tabulky
- Shift Row – Bajty jsou uspořádány do matice a posunovány
- Mix Column – Jednotlivé sloupce matice jsou násobeny konstantní maticí
- Add Round key – Každý bajt je zašifrován bajtem subklíče s využitím logické operace XOR

1.2 RSA

K asymetrickému šifrování je zde využita šifra RSA[2], která funguje na principu faktorizace čísel. V případě asymetrických šifer se používají dva různé klíče – soukromý a veřejný klíč. Klíče jsou generovány následovně:

- Zvol náhodně dvě veliká prvočísla p a q a vypočti n, kde $n = p * q$. n je veřejný modulus.
- Spočti $\phi(n)$ takové, že $\phi(n) = (p-1) * (q-1)$
- Zvol veřejný exponent e takový, že $2 < e < \phi(n)$ a $gcd(e, \phi(n)) = 1$, tedy e a $\phi(n)$ jsou nesoudělná čísla.
- Spočti soukromý exponent d tak, že $d \equiv e^{-1}\ mod\ \phi(n)$, tedy d je výsledkem multiplikativní inverze $e\ mod\ \phi(n)$
- Soukromý klíč tvoří dvojice (n, d) a veřejný (n, e)

[1] `https://en.wikipedia.org/wiki/Advanced_Encryption_Standard`
[2] `https://en.wikipedia.org/wiki/RSA_(cryptosystem)`

Existují dva způsoby použití – šifrování a elektronický podpis. Šifrování a dešifrování probíhá následovně:

$$c = m^e \ mod \ n$$

$$m = c^d \ mod \ n$$

Kde m je původní zpráva, e je veřejný exponent, d je soukromý exponent, n je veřejný modulus a c je výsledný text.

Elektronický podpis s se pak počítá (s využitím zmíněných hodnot) následovně:

$$s = m^d \ mod \ n$$

$$m = s^e \ mod \ n$$

2 Použití klíčů

V projektu je jako počáteční klíč pro šifru AES zvolena náhodná posloupnost 16 bytů ze souboru $/dev/urandom$ a klíč má tedy 128 bytů. Dvojice asymetrických RSA klíčů je pak generována s využitím knihovny *Crypto*[3] a soukromý klíč má délku 2048 bitů. Generování asymetrických klíčů je časově složitá operace, proto jsou generovány pouze tehdy, když se ve složce $/cert$ nenacházejí.

Nultým krokem, než mohou být klíče použity, je výměna veřejných klíčů mezi klientem a serverem. Ta se provádí zasláním klíče přes otevřený socket druhé straně.

Aby byla výměna naprosto korektní, bylo by nutné ze strany klienta si ověřit, že veřejný klíč serveru je validní, tedy server je tím, za koho se vydává, což se provede kontrolou tzv. chain of trust, kdy jednotlivé veřejné klíče jsou podepsané soukromým klíčem vyšší instance. Toto ovšem mé řešení neprovádí z důvodu, že žádná známá vyšší instance neexistuje.

Prvním krokem v použití klíčů je podepsání vygenerovaného hashe zprávy soukromým klíčem odesílatele, přičemž hash je ještě předtím rozšířen sekvencí bitů $11111...100$ až na délku 2047 bitů. Sekvence je přidána na začátek hashe, což se provádí za účelem zvýšení bezpečnosti a zabránění *chosen plaintext* útoku. Podpis je přidán za původní zprávu. Tento podpis zajišťuje integritu zprávy.

Následuje vygenerování náhodného AES klíče, který je použit k zašifrování odesílané zprávy. Poté je obdobně jako hash rozšířen, přidán ke zprávě a zašifrován

[3] `https://pycryptodome.readthedocs.io/en/latest/src/introduction.html`

veřejným klíčem příjemce. Jednotlivé části zprávy jsou pro možnost dekódování odděleny apostrofy ($'$).

Odpověď ohledně úspěchu/neúspěchu server klientovi oznámí zprávou, ke které přiloží původní hash klientovy zprávy, zprávu zašifruje novým AES klíčem, který rozšíří a zašifruje veřejným klíčem příjemce a ke zprávě přiloží.

Příjemce zprávu dekóduje a pokud hash odpovídá, vypíše odpověď serveru, jinak byla narušena integrita i této zprávy.

3 Bezpečnostní analýza

Řešení využívá standardní 128 bytové AES klíče. Ty jsou ovšem před připojením k samotné zprávě rozšířeny až na 2047 bitů. K tomu dochází proto, aby byla zvýšena bezpečnost a útočník musel za účelem dešifrování namísto hádání 1024b (128 bytů je 1024 bitů) hádat 2047b, čímž se výrazně zvyšuje bezpečnost.
Integrita zpráv je ověřována, jak již bylo stručně popsáno, na základě odpovědi ze strany serveru. Server ke svojí odpovědi přiloží hash zprávy, který mu klient poslal. Obdobně jako klient vygeneroval svůj AES klíč, rozšířil jej, zašifroval a přiložil ke zprávě, tak i server provede tento proces, čímž zajistí, že tuto zprávu nebude možné bez znalosti dešifrovacího asymetrického klíče rozluštit. Na straně klienta pak přesne k tomuto rozluštění dojde a v případě, že hash odpovídá té zprávě, kterou klient odeslal, je vypsána odpověď serveru potvrzující přijetí a nenarušení integrity.

Pokud k narušení integrity došlo už při cestě od klienta k serveru, server to v přiložené odpovědi oznámí a na straně klienta dochází ke kontrole, že daná zpráva přesně odpovídá zprávě, kterou má server v takovém případě odpovědět. Klient je požádán, aby zprávu zaslal znovu, neboť byla narušena integrita.

Jestliže byla integrita narušena u odpovědi serveru samotné, tedy hash původní zprávy neodpovídá a zároveň se nejedná o domluvenou chybovou odpověď, indikuje to právě narušení integrity a klient je o tom informován.

4 Závěr

Testování bylo provedeno na několika zprávách, kde byla ověřena základní funkčnost, přičemž odeslané zprávy byly analyzovány také s využitím programu Wireshark[4]. Výjimečně se zpráva odešle, ovšem dojde k *narušení integrity*, tedy že se

[4] `https://www.wireshark.org/`

nepovede správně ověřit pravost hashe, přestože k žádnému narušení dojít nemělo – v takovém případě by mělo stačit zprávu poslat znovu.