ATIVIDADE 2 RELATORIO

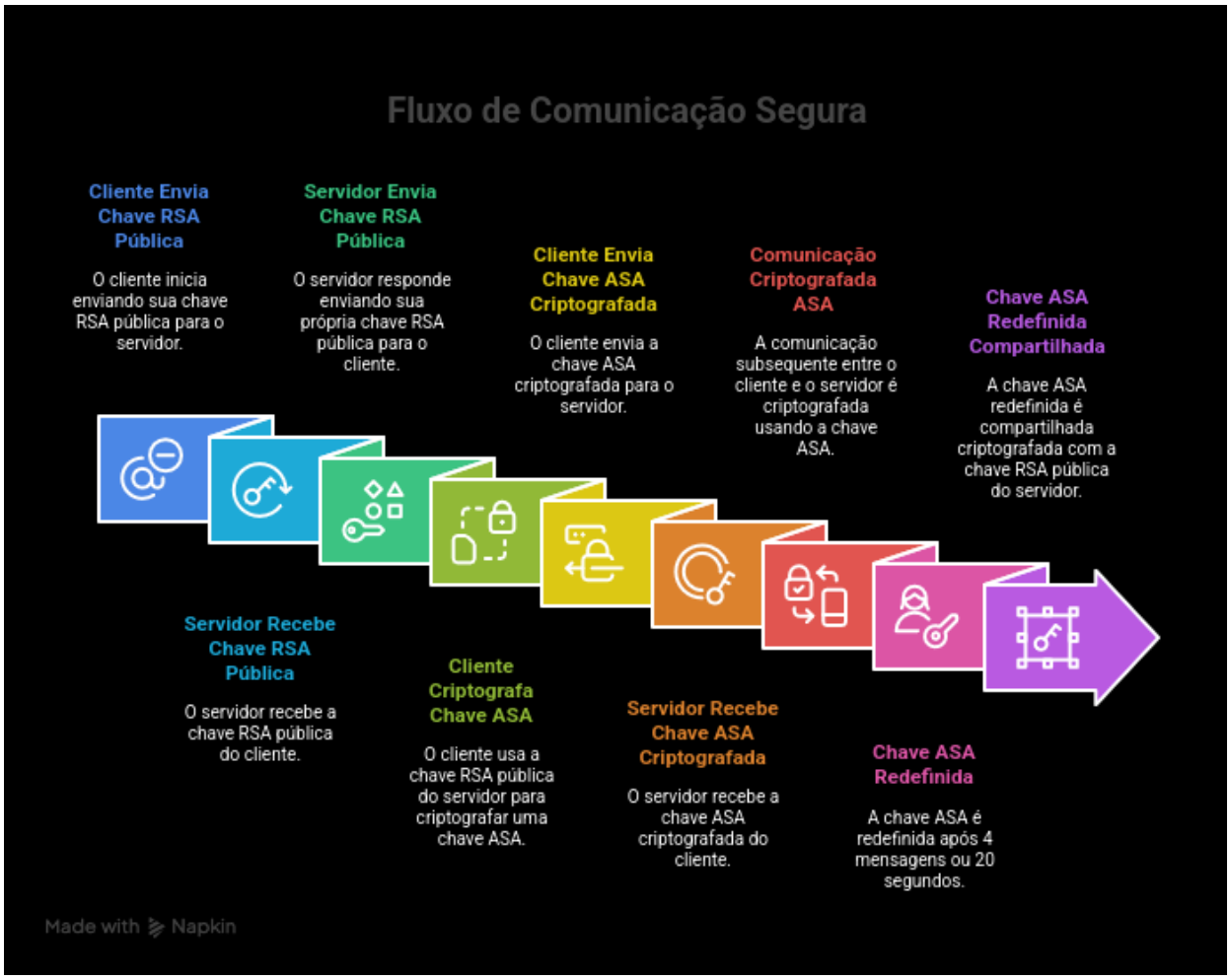
Alunos: Carlos Eduardo e Enzo Zanatta

Sistema desenvolvido:

Desenvolvemos um sistema cliente servidor. O servidor inicia e espera o cliente começar a comunicação. O cliente inicia mandando a public Key dele, e o servidor responde com a public Key dele. Ao receber a public Key do servidor, o cliente gera uma chave AES e criptografa ela com a chave pública do servidor e envia ela para ele. A partir daí o cliente começa a mandar mensagens criptografadas pela chave AES que ele gerou. As mensagens vêm de um arquivo texto, onde elas estão pré definidas. O servidor usa a mesma chave para descriptografar a mensagem recebida do cliente e responde para o cliente confirmando a recepção da mensagem e informando qual o número da mensagem. Após 4 mensagens ou 20 segundos o cliente envia uma nova chave AES para o servidor.

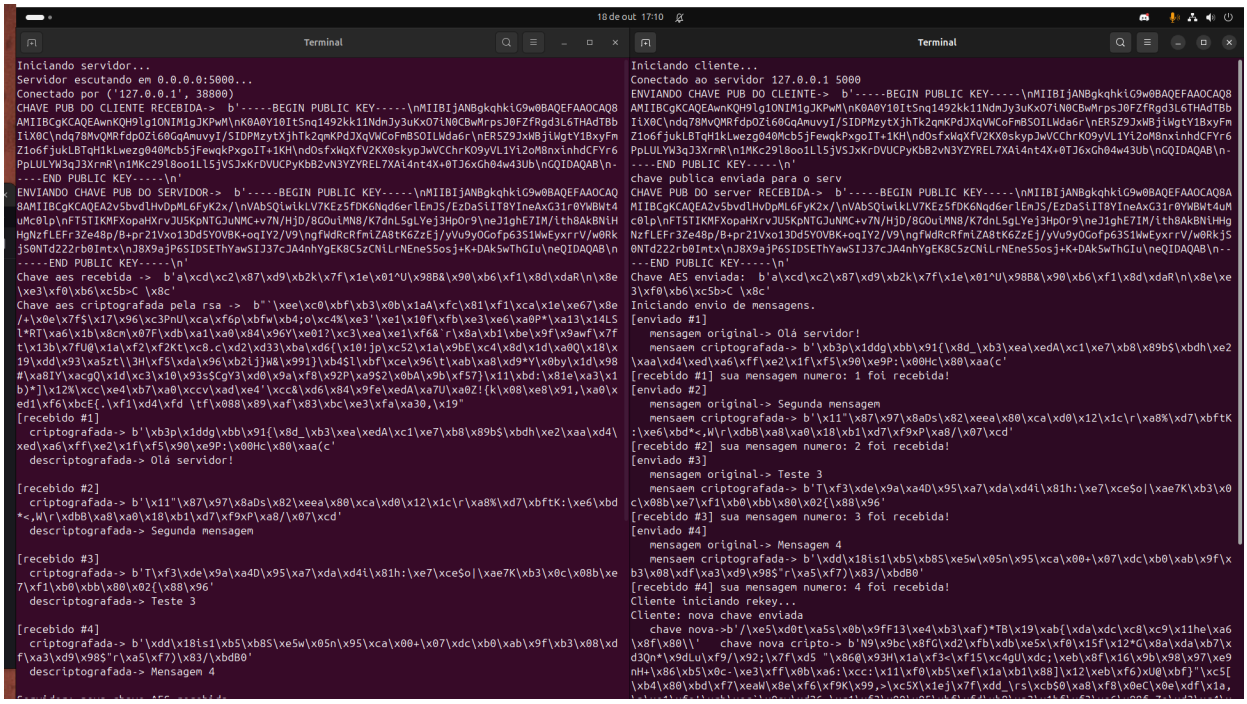
As mensagens trocadas entre eles são do tipo json para que o servidor possa identificar se está recebendo uma mensagem normal ou uma nova chave AES.

Comunicação Segura:



Prints de execução:

INICIO GERAÇÃO DE CHAVES E TROCA DE MENSAGENS



REKEY:

Código fonte

Código fonte Será enviado junto com o pdf.

Análise conceitual

Diferença entre criptografia simétrica, assimétrica e híbrida.

A Criptografia Simétrica usa a mesma chave para cifrar e decifrar, ela é extremamente rápida e eficiente podendo cifrar grandes volumes de dados em milissegundos. O problema principal é que é necessário fazer a distribuição segura da chave e se alguém interceptar a chave simétrica, todo o conteúdo pode ser decifrado. Na nossa implementação, após o cliente e servidor trocarem as chaves RSA, eles usam AES-GCM (criptografia simétrica) para todas as mensagens do chat, porque isso é muito mais rápido e leve.

A Criptografia Assimétrica usa duas chaves diferentes: uma pública e uma privada. Tudo é cifrado com a pública e só pode ser decifrado com a privada (e vice-versa). É segura, pois a chave pública pode ser divulgada sem risco. Mas é muito mais lenta e se torna inadequada para cifrar mensagens grandes ou dados em tempo real. Na nossa implementação, RSA é usado apenas para proteger a troca da chave simétrica AES.

E a Criptografia Híbrida combina as duas anteriores, A assimétrica (RSA) é usada só para trocar uma chave simétrica temporária (AES). Depois, a simétrica faz a cifragem das mensagens reais. Na nossa implementação, o cliente e o servidor começam trocando chaves RSA(cliente gera uma chave AES aleatória → cifra com a RSA do servidor → e envia). A partir daí, todas as mensagens são cifradas com AES. Essa combinação é eficiente e segura, pois a RSA protege a troca inicial da chave (confidencialidade e autenticidade) e a AES garante alto desempenho no envio contínuo de dados.

Por que a abordagem híbrida é mais eficiente?

Porque, o sistema mantém a segurança do RSA sem o custo de desempenho de usar RSA em todo o tráfego.

Como o tempo de expiração contribui para a segurança?

Ele limita o impacto de uma chave comprometida, se alguém conseguisse capturar uma chave AES, só teria acesso às mensagens daquele ciclo. Após o rekey, a chave antiga se torna inútil.

Além de evitar ataques de análise de padrões, reusar a mesma chave AES por muito tempo pode permitir inferências sobre o tráfego, renovando frequentemente a chave, causando uma quebra de padrões estatísticos e reforçando a segurança.

Conclusão e dificuldades encontradas.

Com essa implementação foi possível desenvolver um sistema muito parecido com o TLS, portanto nos agregou um excelente conhecimento prático que vai melhorar muito a segurança de aplicações desenvolvidas por nós no futuro.
Em termos de dificuldade, tivemos a questão de formatar os dados para envio e recepção, muitos erros de encode/decode. E também patinamos um pouco na criação da chave síncrona dentro desse contexto da implementação. Principalmente na parte da expiração, e conseguir suportar mensagens com contextos diferentes (mensagem padrão, mensagem de rekey e de compartilhamento da chave pública).