R.S.A.

Introdução

Até a década de setenta, criptografia entre Alice e Bob era sempre feita por combinar um chave secreta K entre eles e usar um par de funções $e_K(m)$, $d_K(m)$ capaz de encriptar e decriptar, respectivamente, uma mensagem m. Nesses sistemas—chamados de criptossistemas clássicos—, $e_K(m)$ e $d_K(m)$ eram muitas vezes a mesma função ou uma facilmente derivável a partir da outra. A cifra de César é uma ilustração desses criptossistemas. Na cifra de César, Alice e Bob precisam ambos conhecer K e, por isso, diz-se que existe uma simetria entre Alice e Bob—a chave K aparece em ambos os lados. Por isso, esses sistemas são chamados de criptografia simétrica. Na criptografia de chave pública, existe uma assimetria entre a chave de encriptação e decriptação e, por isso, são chamados de criptossistemas assimétricos.

Uma grande desvantagem da criptografia simétrica é que ela exige um acordo prévio sobre que chave secreta será usada. Alice, por exemplo, não pode mandar um e-mail a Bob dizendo—usemos a chave "super segredo" porque a Internet não é um canal seguro. Alice teria que talvez pegar um ônibus até a casa de Bob e entregar-lhe pessoalmente o segredo. Eles poderiam estar em países diferentes e outras complicações—talvez Alice não possa viajar ou não possa sair de seu país *et cetera*.

A novidade que os criptossistemas de chave pública trazem é que não é preciso um canal seguro pra se usar criptografia. A ideia essencial é produzir uma forma e_K de encriptar da qual seja computacionalmente inviável obter a chave inversa d_K de decriptação a não ser que você esteja de posse de uma informação privilegiada. Assim, diz-se que e_K é a chave pública e d_K, a chave privada. Um usuário A desse tipo de sistema produz sua chave pública e_A e a distribui entre amigos e público em geral. Já a chave d_A é mantida em segredo. Quando B quer contar um segredo a A, B usa a chave e_A publicada por A. Depois de encriptada, só a chave d_A de A será capaz de decriptar a informação.

Assim, a chave pública de uma pessoa P é uma espécie de cadeado público que qualquer um pode usar. Uma mensagem escrita e trancada numa caixa forte por esse cadeado não poderá mais ser aberta exceto por P porque só P possui o segredo que abre o cadeado.

Se A pretende encriptar uma mensagem pra B, é importante que A obtenha a chave pública correta de B. Pra ter certeza que a chave pública está correta, outros métodos são necessários — como o uso de certificados, assunto que ainda veremos neste curso.

A ideia de criptossistemas de chave pública foi apresentada [1] por Hellman e Diffie em 1976. No ano seguinte, Ronald Rivest, Adi Shamir e Leonard Adleman publicaram [4] o criptossistema RSA (com ajuda de Michael O. Rabin) baseado na dificuldade que a computação clássica tem de fatorar números inteiros. Desde então, vários criptossistemas novos foram publicados. Por exemplo, Taher El Gamal publicou [2] em 1985 um criptossistema baseado na dificuldade que a computação clássica[1] tem de computar um logaritmo discreto.

É importante notar que criptossistemas assimétricos são usualmente mais custosos—em quantidade de operações computacionais—que criptossistemas simétricos e, por isso, na prática não se usa um criptossistema assimétrico pra encriptar grandes quantidades de dados num tempo curto. Numa típica conexão criptografada de Internet, por exemplo, é comum usar a criptografia assimétrica pra trocar um segredo entre interlocutores e em seguida usar um sistema simétrico como o AES pra se comunicar com mais velocidade. Um sistema assim é classificado como *criptografia híbrida*.

Antes de Hellman e Diffie, a ideia de criptografia de chave pública já tinha sido proposta por James H. Ellis—em janeiro de 1970—num artigo com título "*[t]he possibility of non-secret encryption*", sendo *non-secret* um sinônimo de "chave pública". James Ellis trabalhava na inteligência britânica e sua descoberta foi mantida em segredo até 1997. Logo depois de James Ellis, Clifford C. Cocks—colega de James Ellis—publicou um breve artigo de título "*[a] note on non-secret encryption*" em que ele apresenta um criptossistema essencialmente igual ao RSA.

Também antes de Hellman e Diffie, Malcomm J. Williamson—também colega de James Ellis—publicou dois artigos de títulos "*[n]on-secret encryption using a finite field*" e "*[t]houghts on cheaper non-secret encryption*", respectivamente em 1974 e 1976, em que ele mostra como é possível trocar um segredo usando-se um canal inseguro, o que hoje conhecemos como Protocolo Diffie-Hellman. Também mantido em segredo, esses artigos só vieram a público em 1997. Cabe notar que a inteligência estadunidense também conhecia [3] as descobertas porque eles tiverem acesso às publicações britâncias (quando ainda em segredo).

Como gerar as chaves pública e privada?

1. Sorteie dois primos p, q grandes de tamanhos similares. (Por exemplo, sorteie dois primos de aproximadamente 1024 bits cada um.) Compute o produto $n = pq$.

2. Compute $\varphi(n) = \varphi(p)\varphi(q) = (p-1)(q-1)$.

3. Sorteie um exponente e até que e satisfaça $2 < e < \varphi(n)$ e $\text{mdc}(e, \varphi(n)) = 1$.

4. Publique sua chave pública (n, e). (Por exemplo, publique em sua *homepage*.)

5. Usando e e $\varphi(n)$, compute d tal que $de \equiv 1 \bmod \varphi(n)$. O número d é sua chave privada — guarde-a em segredo. (Por razões didáticas, destrua p, q e $\varphi(n)$.

[1] A computação quântica consegue fatorar e computar logaritmos discretos de forma eficiente. Esses criptossistemas, portanto, estão severamente ameaçados pela computação quântica.

Tecnicamente, não precisamos mais deles, embora na prática p, q são úteis pra acelerar a descriptação, fazendo uso do Teorema Chinês do Resto—um resultado que estudaremos no curso.)

Como encriptar uma mensagem $m \bmod n$ para Alice?

Obtenha a chave pública (n, e) de Alice de forma segura. Seja m um inteiro representando a mensagem a ser encriptada, satisfazendo $0 \leq m < n$. Use o *square-and-multiply* pra computar $c := m^e \bmod n$. Assim, m é a mensagem e c é a encriptação de m. Envie c a Alice.

Como Alice deve decriptar $c \bmod n$?

Alice usa o *square-and-multiply* pra computar $c^d \bmod n = m$.

Terminologia

A mensagem encriptada c é chamada de *ciphertext*, que é termo técnico usado pra representar o texto cifrado. A mensagem pura m é chamada de *plaintext*. A palavra *plain* se traduz pra "puro" ou "simples". O termo *plain text* usualmente significa "texto puro"—um texto simples, sem formatações especiais, sem uma estrutura interna de alta precisão.

> All I can say is that my life is pretty plain
> I like watching the puddles gather rain
> – Brad Smith, 1993.

É assim que se usa RSA na prática?

Não. Observe, por exemplo, que a mensagem $m = 0$ ou $m = 1$ seria sempre transformada para o próprio $c = 0$ ou $c = 1$, respectivamente, o que não ocorre na prática. O que descrevemos acima é às vezes chamado de *plain* RSA. Na prática—ou seja, em uma implementação real de RSA—, o sistema é mais complicado. Por exemplo, números m pequenos não são encriptados de forma isolada como ilustramos acima. A encriptação é feita em blocos de tamanho fixo, contendo vários desses números isolados como se fossem um único número grande — ou seja, concatena-se os bytes que se deseja encriptar até completar um bloco. O tamanho de um bloco, a estratégia de construção desses blocos *et cetera* são definidos em documentos tipicamente chamados de "padrão". Por exemplo, existe um documento publicado em 2016 chamado `RFC 8017` — em sua versão 2.2 até novembro de 2023 —, que descreve a especificação usada na prática por RSA hoje. São exatamente esses detalhes que tornam complicada a implementação de criptossistemas como o RSA, mas são detalhes inevitáveis porque o sistema não é seguro sem eles. Se desejássemos compreender a prática, teríamos mais trabalho pela frente.

Uma mensagem encriptada é garantidamente íntegra?

Não. Quando Bob encripta uma mensagem pra Alice, ora nenhuma Bob usa qualquer informação que seja vinculada a ele próprio. Por exemplo, Bob não precisa nem mesmo ter uma chave privada RSA pra encriptar alguma coisa pra Alice. Bob usa só a chave pública de Alice pra encriptar mensagens pra Alice. Assim, as mensagens encriptadas que Alice recebe não são vinculadas a seus autores. Encriptação é uma técnica que garante *confidencialidade*, mas não garante *autenticidade* e nem *integridade*. Por exemplo, se Bob entrega ao carteiro uma carta encriptada pra Alice, o carteiro pode tranquilamente entregar a Alice uma outra carta (talvez escrita por ele próprio) e Alice não terá como saber se a carta foi adulterada no meio do caminho—a integridade da carta não pôde ser mantida.

Referências

[1] Whitfield Diffie e Martin Hellman. "New directions in cryptography". Em: *IEEE transactions on Information Theory* 22.6 (1976), pp. 644–654. URL: https://bit.ly/2P19sSU.

[2] Taher ElGamal. "A public key cryptosystem and a signature scheme based on discrete logarithms". Em: *IEEE transactions on information theory* 31.4 (1985), pp. 469–472.

[3] Tom Espiner. *GCHQ pioneers on birth of public key crypto*. ZDNet. Out. de 2010. URL: https://bit.ly/2GcJn0v.

[4] Ronald L. Rivest, Adi Shamir e Leonard Adleman. "A Method for Obtaining Digital Signatures and Public-key Cryptosystems". Em: *Commun. ACM* 21.2 (fev. de 1978), pp. 120–126. ISSN: 0001-0782. DOI: 10.1145/359340.359342. URL: https://goo.gl/J9uoRP.