ICMC USP SÃO CARLOS

GANESH

Ataques às cifras modernas

Bit Flipping e Many-Time Pad

Relembrando...

- One-Time Pad
  - $c = E(k, m) = k \oplus m$
  - $m = D(k, c) = k \oplus c$
  - *Perfect Secrecy*
- Stream-Ciphers
  - $PRG(k) = S$
  - $c = E(S, m) = S \oplus m$
  - $m = D(S, c) = S \oplus c$

Bit Flipping

- Consiste em alterar o conteúdo do *plaintext* sem quebrar a cifra
  - Não é um ataque à cifra, mas sim ao texto
  - Possuo $c_1 = m_1 \oplus k$ (k desconhecido)
  - Desejo $c_2 = m_2 \oplus k$
- $c_1 \oplus m_1 \oplus m_2$
  - $= (m_1 \oplus k) \oplus m_1 \oplus m_2$
  - $= (m_1 \oplus m_1) \oplus k \oplus m_2$
  - $= 0^n \oplus k \oplus m_2$
  - $= k \oplus m_2 = c_2$

Bit Flipping

- O destinatário do *ciphertext* não sabe da alteração
- Útil em mensagens que possuem um padrão conhecido
  - Protocolos públicos
  - Onde é possível deduzir informações
- Caso a mensagem original não seja conhecida, em alguns casos também é possível utilizar essa técnica
  - Situações onde o conteúdo do *plaintext* altera algum estado do *software*


Demonstração

Bit Flipping – como evitar?

- Garantir o segundo pilar da criptografia
  - Integridade
- Calcular o *hash* do *plaintext* e enviar junto com o *ciphertext*
- Utilização de um MAC (*Message authentication code*)
  - Garante integridade e autenticação
  - Por exemplo, o HMAC

Many-Time Pad

- Utilização (errada) do One-Time Pad ou de Stream-Ciphers para encriptar múltiplas mensagens com a mesma chave
  - Permite extrair padrões das mensagens
- Suponha m_1 e m_2 encriptados com a chave k
  - $c_1 = m_1 \oplus k$ e $c_2 = m_2 \oplus k$
  - $c_1 \oplus c_2 = m_1 \oplus k \oplus m_2 \oplus k = m_1 \oplus m_2$

Many-Time Pad

- Existem, pelo menos, 2 métodos conhecidos de ataque a essa vulnerabilidade
  - Utilização de um *Crib*
  - Encontrar todos os prováveis espaços das mensagens

Many-Time Pad - *Crib*

- Possuo $c_1 \oplus c_2$
- Suspeito que um *crib* p esteja presente em m_1 ou m_2
- Obtenho o resultado de $c_1 \oplus c_2 \oplus p$
- Se o resultado corresponder a um *plaintext* plausível, obtive sucesso
  - Utilizo o contexto do *plaintext* obtido anteriormente para deduzir um novo *crib*
- Mais eficiente caso possua poucos *ciphertexts*


Demonstração

Many-Time Pad - Espaços

- Tabela ascii
  - [A-Z] = [65-90] = (010xxxxx)
  - [a-z] = [97-122] = (011xxxxx)
  - Espaço = 32 = (0010 0000)
  - ‘a’ - ‘A’ = 32

Many-Time Pad - Espaços

- [A-Z] ⊕ Espaço = [97-122]

  011xxxxx

  ⊕ 00100000

  = 010xxxxx

Many-Time Pad - Espaços

- [a-z] ⊕ Espaço = [65-90]

  010xxxxx

  ⊕ 00100000

  = 011xxxxx

Many-Time Pad - Espaços

- Tendo interceptado um conjunto C de n *ciphertexts*, obtemos $m_i \oplus m_j$, $\forall\, m_i \neq m_j \in C$
- Os bytes de $m_i \oplus m_j$ no intervalo [65-90] ou [97-122] são possíveis espaços
- Queremos as posições cuja frequência de possíveis espaços seja próxima de n

Many-Time Pad - Espaços

- Se m_k possui um espaço na posição p, obtemos essa posição da chave
  - $k[p] = C_k \oplus$ ‘espaço’
- Mais eficiente caso possua vários *ciphertexts* (automatização)


Desafio

GANESH

Grupo de Segurança da Informação
ICMC / USP - São Carlos, SP
http://ganesh.icmc.usp.br/
ganesh@icmc.usp.br