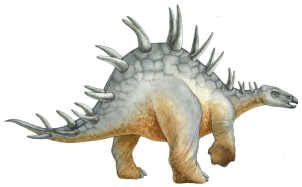
Capítulo 15: Segurança

Capítulo 15: Segurança

- O problema da segurança
- Ameaças ao programa
- Ameaças ao sistema e à rede
- Criptografia como uma ferramenta de segurança
- Autenticação do usuário
- Implementando defesas de segurança
- Uso de firewalls para proteger sistemas e redes
- Classificações de segurança de computador
- Um exemplo: Windows XP



Objetivos

- Discutir as ameaças à segurança e ataques
- Explicar os fundamentos da criptografia, autenticação e hashing
- Examinar os usos da criptografia na computação
- Descrever as diversas contramedidas para os ataques à segurança

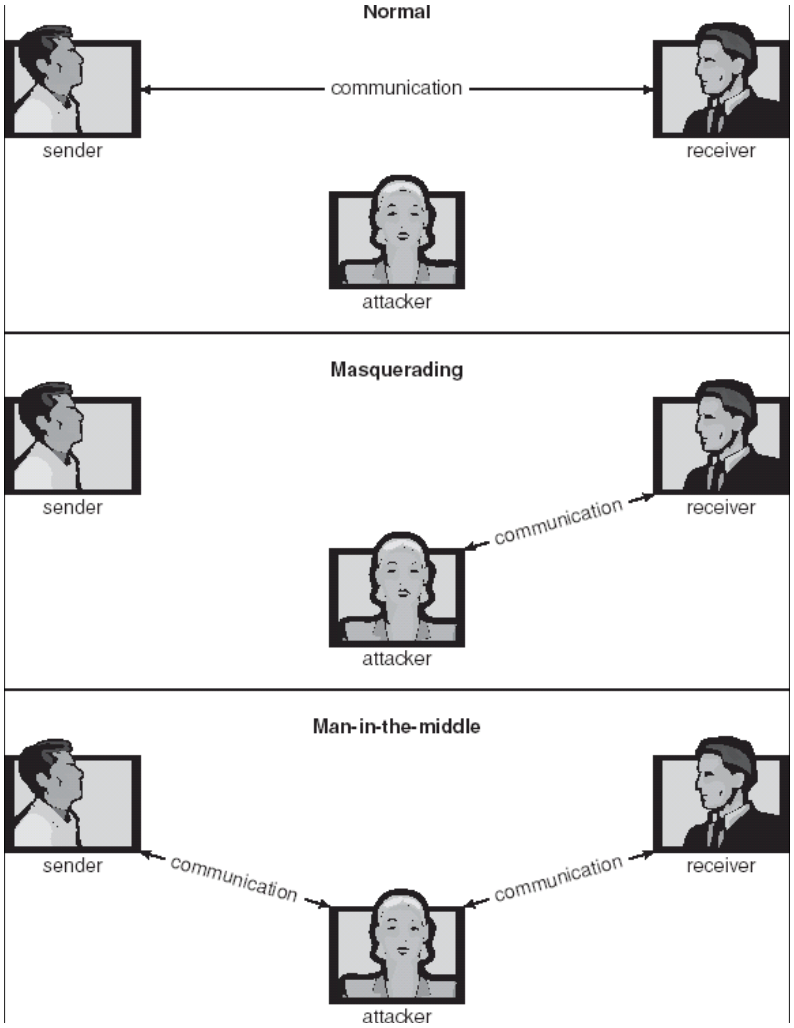
O problema da segurança

- A segurança deve considerar o ambiente externo do sistema e proteger os recursos do sistema
- Intrusos (crackers) tentam quebrar a segurança
- **Ameaça** é violação de segurança em potencial
- **Ataque** é a tentativa de quebrar a segurança
- Ataque pode ser acidental ou malicioso
- É mais fácil proteger contra mau uso acidental do que malicioso

Violações de segurança

- Categorias
  - **Quebra de confidencialidade**
  - **Quebra de integridade**
  - **Quebra de disponibilidade**
  - **Roubo de serviço**
  - **Negação de serviço**

Ataques de segurança padrão



Níveis de medida de segurança

- A segurança deve ocorrer nos quatros níveis para ser eficaz:
  - Físico
  - Humano
  - Sistema operacional
  - Rede
- Um princípio de segurança: Uma corrente é tão fraca quanto seu elo mais fraco

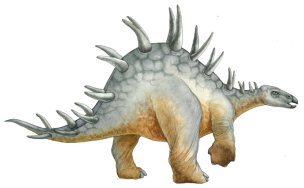
Ameaças ao programa

- Cavalo de Tróia
  - Segmento de código que faz mau uso de seu ambiente
  - **Spyware, janelas popup do navegador, etc...**
- Porta de armadilha
  - Identificador ou senha de usuário específico, que contorna os procedimentos de segurança normais
  - Poderia estar incluída em um compilador
- Bomba lógica
  - Programa que inicia um incidente de segurança sob certas circunstâncias
- Estouro de pilha e buffer
  - Explora um bug em um programa (estoura a pilha ou buffers de memória)

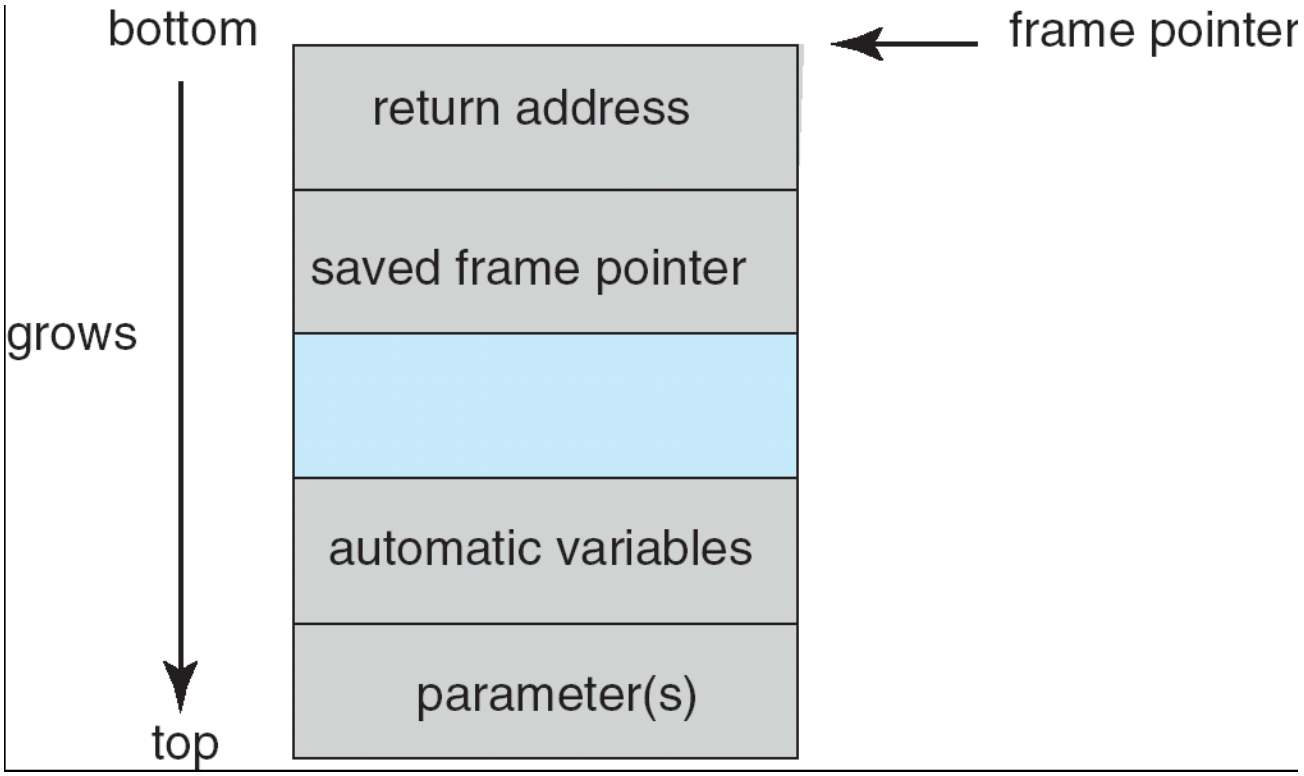
Programa C com condição de estouro de buffer

```
#include <stdio.h>
#define BUFFER_SIZE 256

int main(int argc, char *argv[])
{
   char buffer[BUFFER_SIZE];

   if (argc < 2)
      return -1;
   else {
      strcpy(buffer,argv[1]);
      return 0;
   }
}
```

Se argv[1] for mair que BUFFER_SIZE lascou...



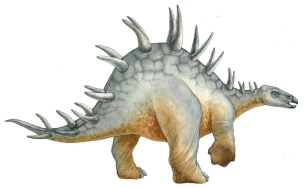
Layout de quadro de pilha típico



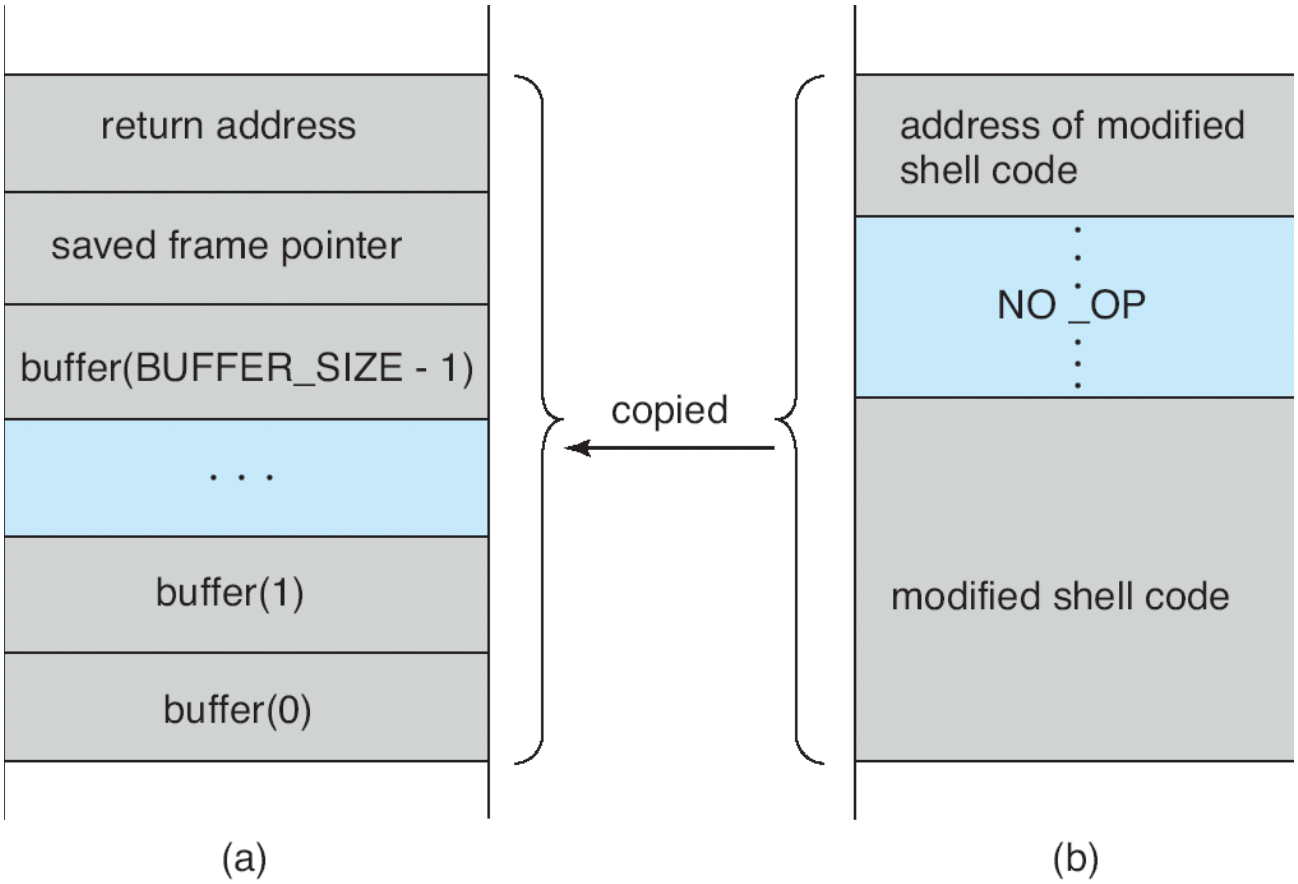
Código shell modificado

```
#include <stdio.h>

int main(int argc, char *argv[])
{
    execvp(``\bin\sh'',``\bin \sh'', NULL);
    return 0;
}
```

- O programa (malicioso) abre o shell com as mesmas permissões que o processo tiver
- O código precisa ser ser curto para caber em um quadro da pilha
- Depois de compilado, o código binário resultante (“exugado”) é colocado no buffer



Quadro de pilha hipotético



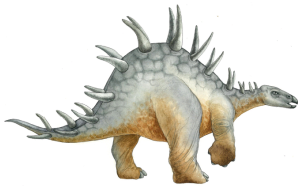
Antes do ataque

Após ataque

Ameaças ao programa (cont.)

- Vírus
  - Fragmento de código embutido no programa legítimo
  - Muito específico à arquitetura de CPU, sistema operacional, aplicações
  - Normalmente, vem de e-mail ou como uma macro
    - Macro do Visual Basic para reformatar disco rígido

```
Sub AutoOpen()
Dim oFS
  Set oFS =
  CreateObject(''Scripting.FileSystemObject''
  )
  vs = Shell(''c:command.com /k format
  c:'',vbHide)
End Sub
```



Ameaças ao programa (cont.)

- **Colocador de vírus** insere vírus no sistema
- Algumas categorias de vírus
  - Arquivo
  - Boot
  - Macro
  - Código fonte
  - Polimórfico (muda toda vez que é instalado)
  - Criptografado
  - Furtivo (modifica partes do sistema para não ser detectado)
  - Tunelamento (se instala na cadeia do tratador de interrupções ou em drivers)
  - Multipartite ou híbridos (infecta boot, memória, arquivos, ...)
  - Blindado (difícil de entender o funcionamento)

Ameaças ao sistema e à rede

- Vermes – se auto-replica (sem necessidade de infectar arquivo legítimo)
- Verme da Internet
  - Explora recursos de rede para se espalhar
  - Programa **gancho de atracação** faz o upload do programa de verme principal
- Varredura de porta
  - Tentativa automatizada de conectar a um grupo de portas em um ou vários endereços IP
- Negação de serviço
  - Sobrecarrega o computador vítima, impedindo que realize qualquer trabalho útil
  - Negação de serviço distribuída vem de vários locais ao mesmo tempo

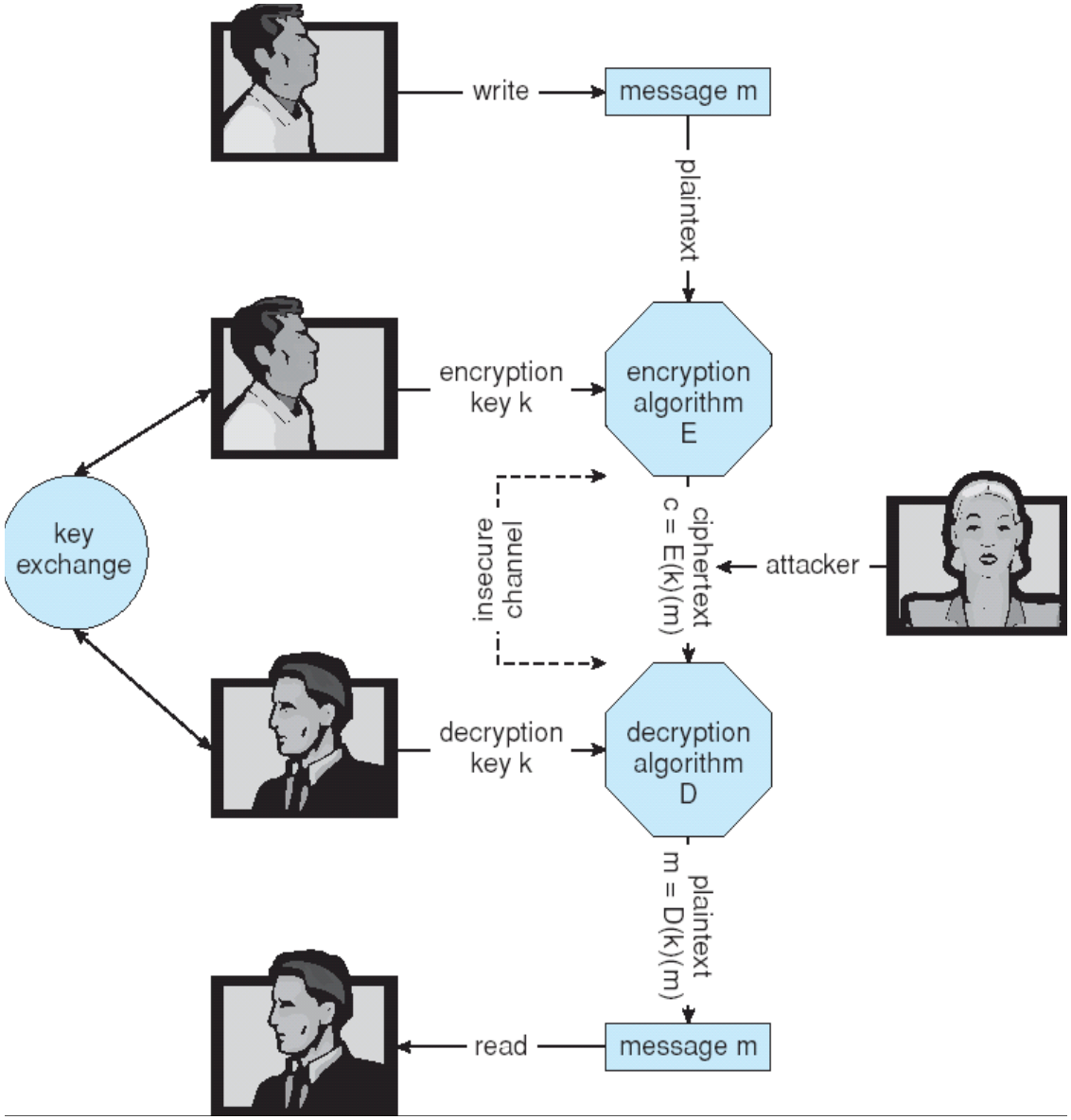
Varredura de porta em Java

```
public class PortScanner
{
  public static final int PORT_MAX = 255;

  public static final int TIMEOUT_VALUE = 1000; // 1 second

  public static void main(String[] args) {
    InetAddress host = InetAddress.getByName(args[0]);

    for (int port = 0; port <= PORT_MAX; port++) {
      try {
        SocketAddress addr = new InetSocketAddress(host,port);
        Socket sock = new Socket();

        // attempt to make a connection to (host + port)
        sock.connect(addr,1000);
        System.out.println("Listening at port: " + port);
        // we could now try to exploit the service
        // listening at this port
      }
      catch (java.io.IOException ioe) {
        // not listening to this port
      }
    }
  }
}
```

Criptografia como ferramenta de segurança

- Principal ferramenta de segurança disponível
  - Origem e destino das mensagens não podem ser confiados sem criptografia
  - Meios de restringir emissores (*origens*) e/ou receptores (*destinos*) em potencial das *mensagens*
- Baseado em segredos (**chaves**)

Comunicação insegura por meio inseguro



Codificação

- Algoritmo de codificação consiste em
  - Conjunto de *K* chaves
  - Conjunto de *M* mensagens
  - Conjunto de *C* textos cifrados (mensagens codificadas)
  - *E*(*k*) é uma função para gerar textos cifrados a partir de mensagens.
  - *D*(*k*) é uma função para gerar mensagens a partir de textos cifrados.

Codificação simétrica

- Mesma chave usada para codificar e decodificar
  - $E(k)$ pode ser derivado de $D(k)$, e vice-versa
- DES é o algoritmo de codificação simétrica em bloco mais utilizado (criado pelo governo dos EUA)
  - Codifica um bloco de dados de cada vez
- Triple-DES considerado mais seguro
- Advanced Encryption Standard (**AES**), **twofish**
- RC4 é a cifra de stream simétrica mais comum, mas com vulnerabilidades conhecidas

Codificação assimétrica

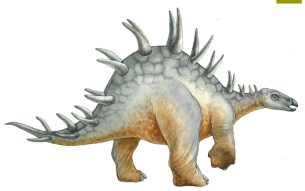
- Codificação por chave pública baseada em cada usuário tendo duas chaves:
  - chave pública – chave publicada usada para codificar dados
  - chave privada – chave conhecida apenas do usuário individual, usada para decodificar dados
- Deve ser um esquema de codificação que possa se tornar público sem deixar fácil a descoberta do esquema de decodificação
  - Mais comum é a cifra por bloco RSA
  - Algoritmo eficiente para testar se um número é primo ou não
  - Nenhum algoritmo eficiente conhecido para encontrar os fatores primos de um número

Criptografia (cont.)

- Nota: criptografia simétrica baseada em transformações, assimétrica baseada em funções matemáticas
  - Assimétrica usa muito mais computação
  - Normalmente não usada para criptografia de dados em massa

Autenticação

- A autenticação restringe o conjunto de emissores (enquanto a codificação restringe o conjunto de receptores) em potencial de uma mensagem
  - Complementar e às vezes redundante à codificação
  - Também pode comprovar que a mensagem não foi modificada (através de hash)
- Componentes do algoritmo
  - Um conjunto *K* de chaves
  - Um conjunto *M* de mensagens
  - Um conjunto *A* de autenticadores
  - Uma função $S : K \rightarrow (M \rightarrow A)$
    - $S(k)$ é uma função para gerar autenticadores de mensagens
  - $V(k)$ é uma função para verificar autenticadores em mensagens



Autenticação – Assinatura digital

- Baseada em chaves assimétricas e algoritmo de assinatura digital (RSA, por exemplo, só que inverte qual é a chave pública/privada)
- Autenticadores produzidos são **assinaturas digitais**

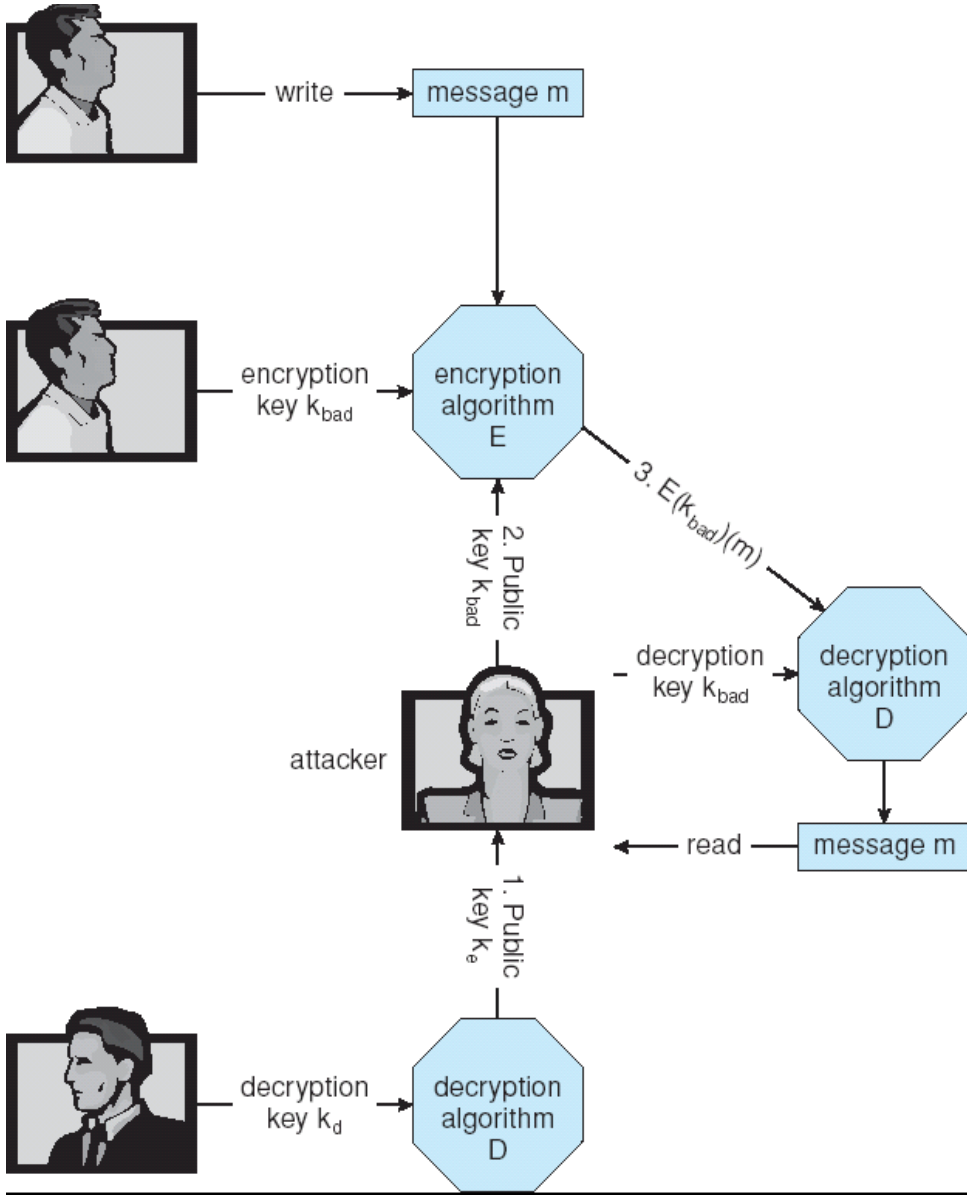
Autenticação (cont.)

- Por que a autenticação é um subconjunto da codificação?
  - Menos cálculos (exceto para assinaturas digitais RSA)
  - Autenticador normalmente mais curto que a mensagem
  - Às vezes, deseja autenticação, mas não confidencialidade
  - Pode ser base para **não-repúdio**

Distribuição de chave

- Entrega de chave simétrica é um imenso desafio
  - Às vezes feito **off-line**
- Chaves assimétricas podem se proliferar – armazenadas em **chaveiro**
  - Até mesmo a distribuição de chave assimétrica precisa de cuidado – ataque do homem no meio

Ataque do homem no meio sobre criptografia assimétrica



Certificados digitais

- Prova de quem ou o que possui uma chave pública
- Chave pública assinada digitalmente por uma parte confiável
- Parte confiável recebe prova de identificação da entidade e certifica que a chave pública pertence à entidade
- Autoridade de certificação é uma parte confiável – suas chaves públicas incluídas com distribuições de navegador Web
  - Elas respondem por outras autoridades assinando digitalmente suas chaves, e assim por diante

Exemplo de criptografia - SSL

- Inserção de criptografia em uma camada do modelo de rede ISO (a camada de transporte)
- SSL – Secure Socket Layer (evoluiu para TLS)
- Protocolo criptográfico que limita dois computadores a só trocarem mensagens um com o outro
- Usado entre servidores e navegadores Web para a comunicação segura (números de cartão de crédito)
- O servidor é verificado com um **certificado**, garantindo que o cliente está falando com o servidor correto
- Criptografia assimétrica usada para estabelecer uma **chave de sessão** segura para o núcleo da comunicação durante a sessão
- Comunicação entre cada computador, então, usa criptografia por chave simétrica

Autenticação do usuário

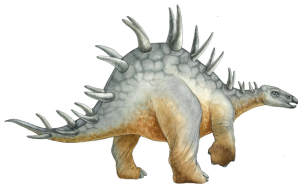
- Crucial para identificar o usuário corretamente, pois sistemas de proteção depende da ID do usuário
- Identidade do usuário normalmente estabelecida por *senhas*, pode ser considerada um caso especial de chaves ou capacidades
  - Também pode incluir algo que o usuário tenha e/ou um atributo do usuário
- Senhas devem ser mantidas secretas
  - Mudança freqüente de senhas
  - Uso de senhas “não-adivinháveis”
  - Log de todas as tentativas de acesso inválidas
- Senhas também podem ser codificadas ou ter permissão para serem usadas apenas uma vez

Implementando defesas de segurança

- **Defesa em profundidade** é a teoria de segurança mais comum – múltiplas camadas de segurança
- Política de segurança
- Avaliação de vulnerabilidade
- Detecção de intrusão
- Proteção contra vírus
- Auditoria, contabilidade e logging de todas as atividades da rede ou específicas do sistema

Uso de firewalls para proteger sistemas e redes

- Um firewall de rede é colocado entre hosts confiáveis e não confiáveis
  - O firewall limita o acesso da rede entre esses dois domínios de segurança
- Problema:
  - Regras de firewall normalmente baseadas no nome de host ou endereço IP, quem pode ser forjados
- **Firewall pessoal** é camada de software em determinado host
  - Pode monitorar/limitar tráfego de/para o host
- **Firewall de proxy de aplicação** entende os protocolos que as aplicações falam pela rede (por exemplo, SMTP)
- **Firewall de chamada de sistema** monitora todas as chamadas do sistema e aplica regras a elas (por exemplo, esse programa pode executar essa chamada do sistema)



Final do Capítulo 15