QoS e segurança

Mestrado em Engenharia de Computadores e Telemática

2023/2024

Qualidade de serviço

Aplicativos baseados em TCP e UDP

Problema: Avaliar TCP

- Por que o TCP funciona *mal* em redes ad hoc/veiculares?
  - Desenvolvido para redes fixas
  - Assume que todas as perdas são devido ao congestionamento
- Muitas variantes de TCP foram propostas
  - Quão bons eles são? Eles são suficientes?
- Há alguma outra alternativa?
  - Os protocolos não-TCP são a solução?

Visão geral dos conceitos de TCP

• TCP convencional: Tahoe, Reno, New-Reno

• A taxa de envio é controlada por

– Janela de congestionamento (*cwnd*): limita o número de pacotes em trânsito

– Limite de início lento (*ssthresh*): quando a prevenção de congestionamento começa

• Detecção de perdas

– 3 ACKs duplicados (mais rápido, mais eficiente)

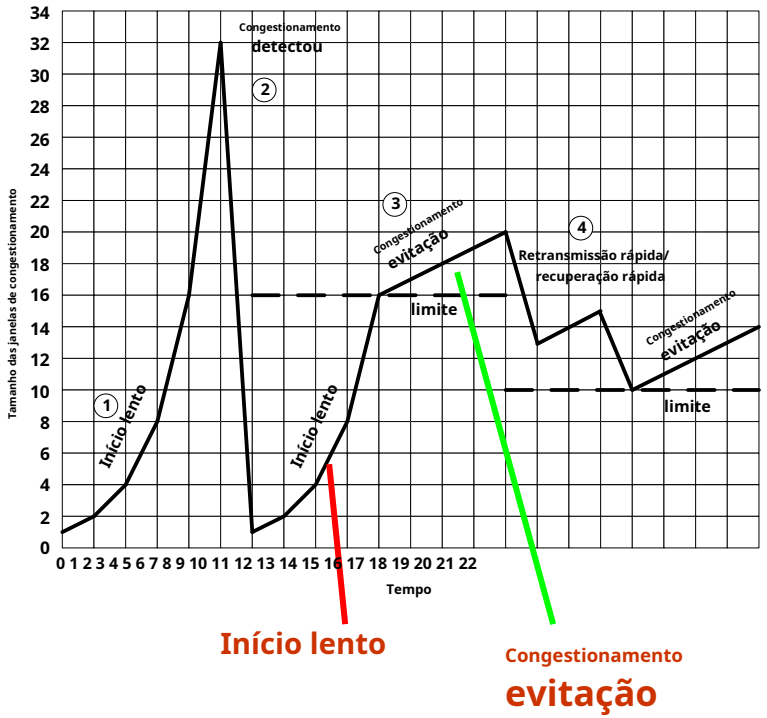
– O temporizador de retransmissão expira (mais lento, menos eficiente)

• Visão geral dos mecanismos de controle de congestionamento

– Fase de arranque lento: *cwnd* começa em 1 e aumenta exponencialmente

– Evitar congestionamento (CA): *cwnd* aumenta linearmente

– Retransmissão rápida e recuperação rápida: acionada por 3 ACKs duplicados



O que há de diferente nas redes sem fio e também nas ad-hoc?

1. Mobilidade
   - Estabilidade e disponibilidade de rotas
2. Alta taxa de erro de bits
   - Pacotes podem ser perdidos devido ao “ruído”
3. Imprevisibilidade/Variabilidade
   - Difícil estimar tempo limite, RTT, largura de banda
4. Contenção: pacotes competem por tempo de antena
   - Contenções intrafluxo e interfluxo
5. Conexões longas têm baixo desempenho
   - A taxa de transferência de mais de 4 saltos cai drasticamente

Por que o TCP funciona menos em ad-redes hoc?

- O TCP interpreta mal as falhas de rota como congestionamento
  - Efeitos: Reduza a taxa de envio
  - Pacotes armazenados em buffer (dados e ACKs) em nós intermediários são descartados
  - O remetente encontra tempos limite
    - Sob desconexão prolongada, uma série de tempos limite podem ser encontrados
- TCP interpreta erroneamente erros sem fio como congestionamento
  - Efeitos: Execução incorreta do controle de congestionamento→ queda de desempenho
  - O canal sem fio é propenso a erros em comparação ao canal com fio
    - Desbotamento, interferência, ruído

Por que o TCP f a eu eu eu eu R enos e em ad-redes hoc?

• O pico de atraso faz com que o TCP invoque retransmissões desnecessárias

- Efeitos: O desempenho cai e existem muitas retransmissões desnecessárias
- Variabilidade: Picos não são incomuns aqui
- Picos prejudicam a estimativa e o ajuste de parâmetros
  - RTO, tamanho da janela, limite de início lento

• Ineficiência devido à perda de pacotes retransmitidos

- Efeitos: o desempenho cai significativamente em ambiente de alta perda
- Perder um *retransmitido* pacote dói
  - O TCP pode se recuperar de uma perda (retransmissão rápida)
- Redes com fio: a taxa de perda de pacotes é baixa
- Aqui, a alta perda de pacotes torna o problema significativo

Novas possibilidades?

TCP-Cúbico (1)

- CUBIC é independente de RTT
- O tamanho da janela é uma função cúbica do tempo t, que é o tempo decorrido desde a última ocorrência de congestionamento

$$W_{CUBIC} = C\left(t - \sqrt[3]{\frac{\beta . W_{max}}{C}}\right)^3 + W_{max}$$

- onde *W_CUBIC* é o *cwnd* tamanho para mecanismo cúbico de controle de congestionamento,
- W_máx, uma *cwnd* tamanho imediatamente antes da última redução da janela,
- C, uma constante predefinida (fator de escala),
- β, uma *cwnd* fator de diminuição de tamanho. A redução do tamanho da janela no momento do evento de perda é

$$W(t) \leftarrow W(t^*)(1-\beta).$$

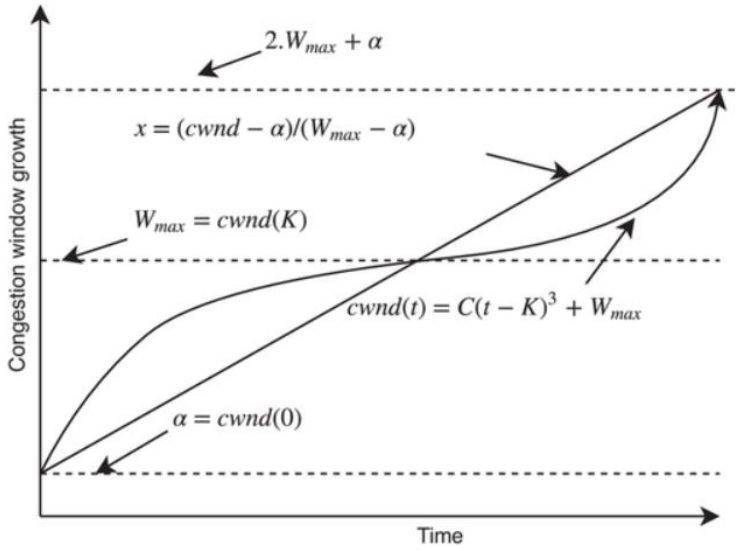
- onde W(t*) é o tamanho cwnd no momento t* da perda de pacotes, ou seja, Wmax.
- Quando uma perda de pacote é detectada, W(t) é reduzido conforme a equação anterior.

https://www.cs.princeton.edu/courses/archive/fall16/cos561/papers/Cubic08.pdf

TCP-Cúbico (2)

• Sempre que ocorre um evento de queda de pacote, o cwnd é reduzido por um fator β, caso contrário, aumentado pelo α para um ACK bem-sucedido. Para TCP com CUBIC, β=0,3, enquanto para QUIC com CUBIC,β=0,3/2=0,15

$$K = \sqrt[3]{\frac{\beta . W_{max}}{C}}$$



RÁPIDO

- O QUIC foi desenvolvido e implantado pelo Google em 2013, mas foi apresentado como padrão em 2021 pela RFC9000.
- Os pacotes QUIC são transportados em datagramas UDP para facilitar melhor a implantação em sistemas e redes existentes.
- O handshake QUIC combina negociação de parâmetros criptográficos (TLS) e de transporte.
  - Está estruturado para permitir a troca de dados de aplicação o mais rápido possível.

– Os protocolos de aplicação trocam informações por meio de uma conexão QUIC por meio de fluxos que são sequências ordenadas de bytes. Dois tipos de fluxos podem ser criados:

- Fluxos bidirecionais, que permitem que ambos os terminais enviem dados.
- Fluxos unidirecionais, que permitem que um único terminal envie dados.

– Evita o bloqueio de cabeçalho em vários fluxos

- Quando ocorre uma perda de pacote, apenas os fluxos com dados nesse pacote são bloqueados, aguardando o recebimento de uma retransmissão, enquanto outros fluxos podem continuar progredindo.

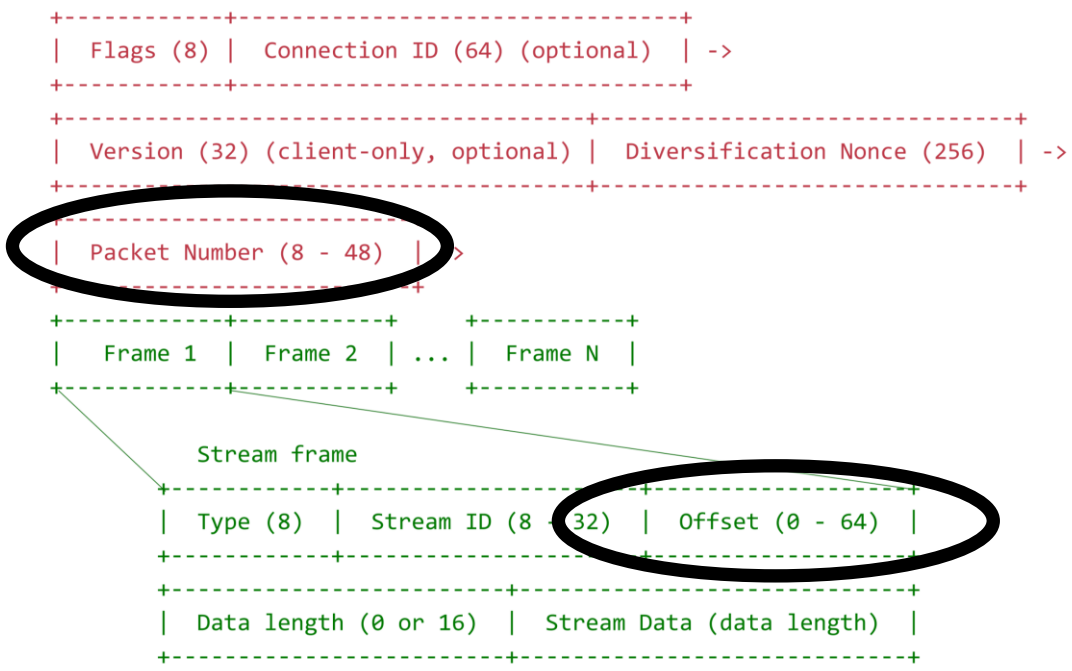
– Dois níveis de controle de fluxo de dados no QUIC:

- Controle de fluxo de fluxo, que evita que um único fluxo consuma todo o buffer de recebimento de uma conexão, limitando a quantidade de dados que pode ser enviada em cada fluxo.
- Controle de fluxo de conexão, que evita que os remetentes excedam a capacidade de buffer do receptor para a conexão, limitando o total de bytes de dados de fluxo enviados em todos os fluxos.

https://datatracker.ietf.org/doc/html/rfc9000

QUIC: estimativa RTT

- Usarnúmeros de sequência separadospara entrega de dados e pacotes
- Dissociar completamente a confiabilidade da entrega solicitada



QUIC: estimativa RTT

- Dissociar completamente a confiabilidade da entrega solicitada
- Os números dos pacotes sãoaumentando monotonicamente(e, portanto, único)
  - Distinto dos deslocamentos de quadro dentro do fluxo
- Use o tempo entre a transmissão de um pacote e seu ACK (identificável usando um número de pacote exclusivo) para estimativa de RTT
  - Independentemente da transmissão original ou retransmissão
- Perda de pacotes detectada por falta de pacotes ACK
- No entanto, o controle de congestionamento é independente do RTT
  - Um dos algoritmos mais utilizados é CUBIC
- O controle de congestionamento não depende dos atrasos do meio sem fio
- O controle de congestionamento é aplicado após a detecção de perda de pacotes
  - Novamente, a perda de pacotes pode não ser um sinal de congestionamento

Classificação de protocolos de transporte

- Como tornar o protocolo de transporte adequado para redes sem fio e ad-hoc?

- Variantes TCP tentam melhorar o desempenho das seguintes maneiras
  - Estimando a largura de banda disponível
  - Explorando a capacidade de buffer

TCP-Vegas

- O TCP Vegas detecta o congestionamento na rede antes que ocorra qualquer perda de pacote e instantaneamente diminui o tamanho da janela. Portanto, o TCP Vegas lida com o congestionamento sem ocorrer perda de pacotes.

- Usos *diff = taxa esperada - taxa real* para regular a taxa de envio

- *Taxa esperada*=tamanho da janela dividido por baseRTT,
- *Taxa real*=tamanho da janela dividido por currentRTT
- baseRTT = É o RTT mínimo medido até o momento.

- *Taxa esperada* é sempre maior que o *Taxa real*

https://cseweb.ucsd.edu/classes/wi01/cse222/papers/brakmo-vegas-jsac95.pdf

TCP-Vegas

- Controle de congestionamento baseado em taxa
  - diff = taxa esperada – taxa real
  - Se diff < a, Vegas aumenta *cwnd* linearmente
  - Se diff > b, Vegas diminui *cwnd* linearmente
  - Se a< diff < b, Vegas mantém *cwnd* inalterado
- Início lento modificado
  - Permite *cwnd* crescer exponencialmente apenas em todos os outros *RTT*
  - Se diff > c, Vegas muda de início lento para prevenção de congestionamento
- Nova retransmissão
  - Lê e registra o tempo de transmissão
  - Quando o DUPACK chega, verifica se está vencido
  - Retransmite sem esperar pelo terceiro DUPACK

Explorando capacidades de buffer

- Capacidade de buffer e informações de sequência (TCP-BuS)
  - – Usa notificação explícita de falha de rota (erro de roteamento) para detectar falhas de rota
  - – Quando ocorre uma falha na rota, os nós intermediários armazenam em buffer os pacotes pendentes e o remetente TCP dobra o tempo limite de retransmissão (*RTO*) valor
  - – Faz uso de mensagens especiais, como consulta localizada e resposta para encontrar um caminho
    - Mensagens modificadas para transportar conexão TCP e informações de segmento
  - – Evite timeouts e retransmissões desnecessárias
  - – Pro: Reduza o número de eventos de tempo limite→reduzir o número de retransmissões
  - – Contra: Requer assistência de nós intermediários
    - Protocolo de roteamento especial é usado

https://ieeexplore.ieee.org/document/6596860

QoS em UDP: compensações

- Em mecanismos do tipo IntServ
  - O aplicativo especifica parâmetros de tráfego e QoS
  - Um protocolo de reserva de recursos estima e reserva recursos suficientes em cada nó no caminho
- Em uma rede sem fio multi-hop, entretanto
  - É difícil estimar os recursos disponíveis
    - Meio compartilhado
      - Todo o tráfego na faixa de transmissão reduz a largura de banda disponível
    - Largura de banda dinâmica devido à mobilidade e contenção de nós
  - É difícil fazer reserva de recursos
    - A reserva de meio compartilhado requer coordenação global
    - Violações podem ocorrer conforme a largura de banda flutua
  - A reserva de recursos está fixada em uma rota
    - Deve ser refeito sempre que houver mudança de rota
    - Isso pode NÃO ser uma boa ideia
      - Tempo para recuperar uma rota e reservar recursos na nova rota

QoS em UDP: compensações

- Em mecanismos do tipo DiffServ
  - – Aplicação escolhe uma classe de serviço
  - – Rede precisa de controle de admissão para evitar sobrecarga de aulas
  - - Exemplos
    - O comportamento de encaminhamento garantido por salto garante a taxa de transferência por salto
    - O comportamento de encaminhamento acelerado por salto garante baixo atraso por salto
- Em uma rede sem fio multi-hop, entretanto
  - – É difícil fazer controle de admissão
    - Os fluxos não passam por nós de entrada comuns
  - – É difícil manter garantias
    - A distribuição do fluxo varia conforme as rotas mudam
    - A largura de banda flutua

Roteamento QoS

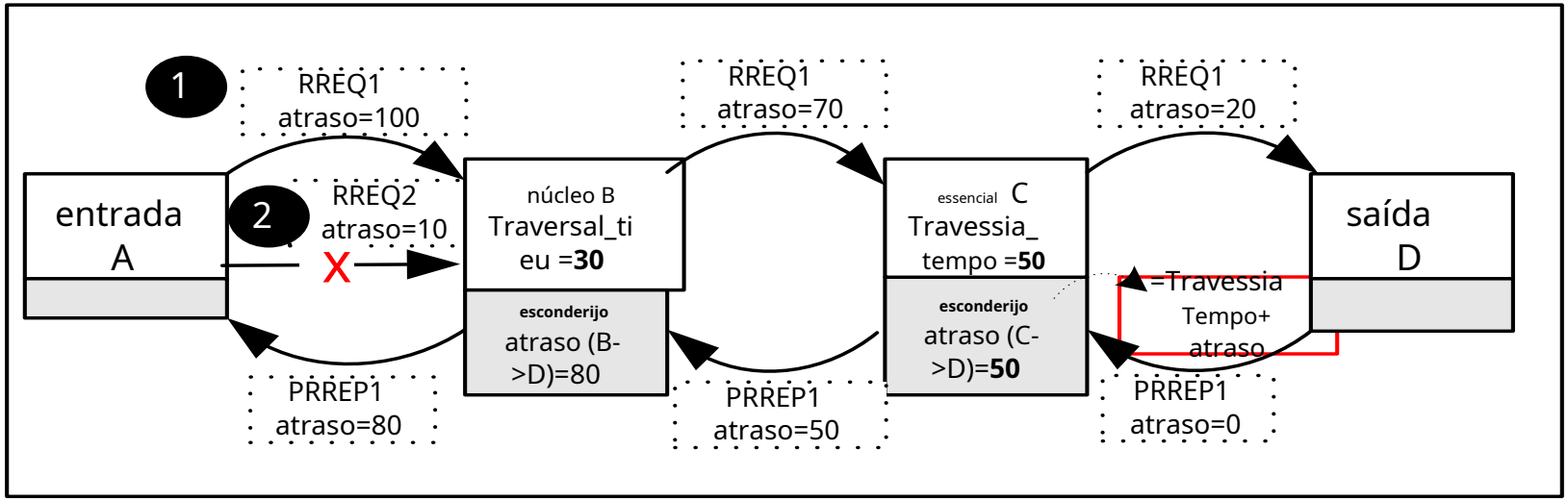
- O roteamento é um componente essencial para QoS
- Pode informar um nó de origem sobre a largura de banda e a disponibilidade de QoS de um nó de destino e do caminho para o nó de destino
- Adicionar requisitos de QoS nas métricas de roteamento
  - Difícil rotear manutenção
  - Sobrecarga de roteamento de QoS
  - Os recursos reservados podem não ser garantidos
  - Precisa ser responsivo à mobilidade dos nós

QoS para AODV

- Adicionar ramais às mensagens de rota (RREQ, RREP)
- O nó que recebe uma extensão RREQ + QoS deve ser capaz de atender ao requisito de serviço para retransmitir o RREQ (se não estiver em cache)
- Para lidar com as extensões de QoS, algumas alterações precisam ser feitas nas tabelas de roteamento
- Campos atuais AODV
  - Número de sequência de destino, Interface, Contagem de saltos, Próximo salto, Lista de precursores
- Novos campos AODV (4 novos campos)
  - Atraso máximo
  - Largura de banda mínima disponível
  - Lista de fontes que solicitam garantias de atraso
  - Lista de fontes que solicitam garantias de largura de banda

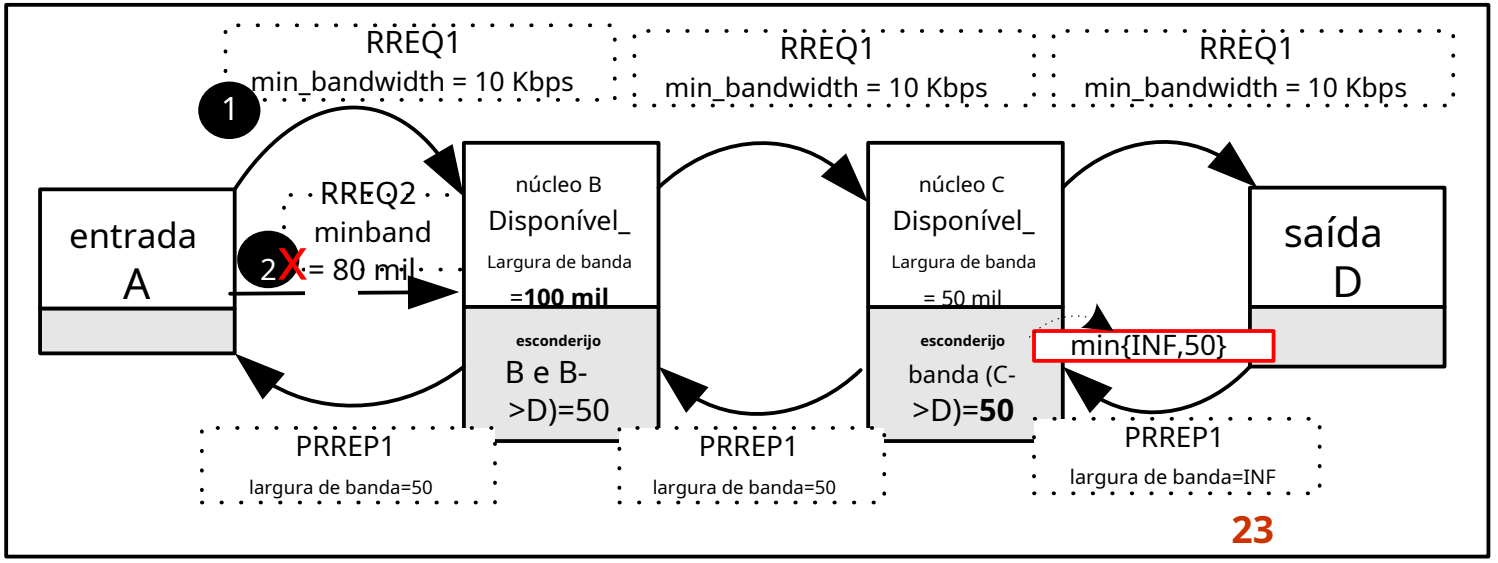
QoS para AODV - Atraso

- Atraso no Tratamento
  - Extensão máxima de atraso
  - Lista de fontes que solicitam garantias de atraso
  - Usado durante o processo de descoberta de rota



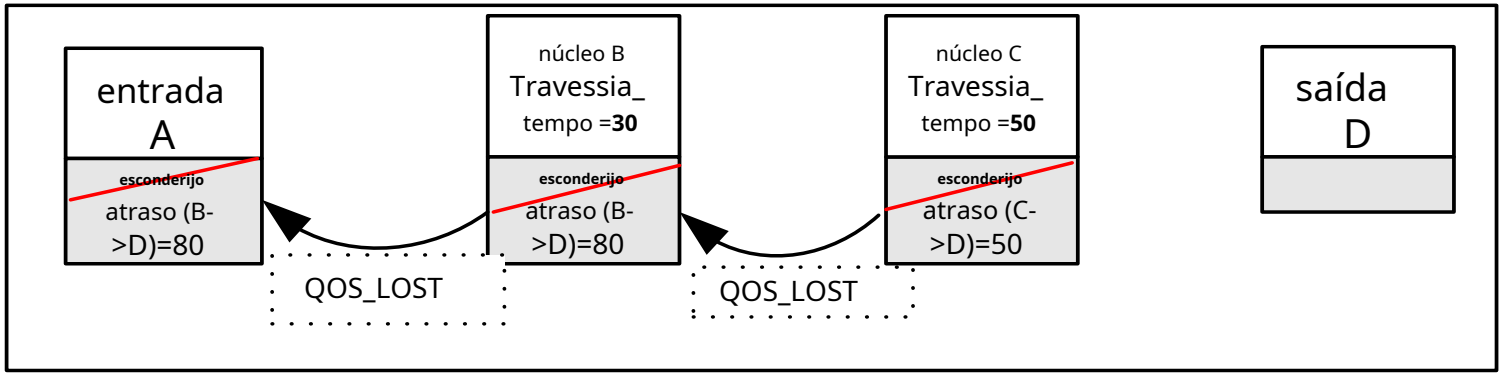
QoS para AODV - largura de banda

- Lidando com largura de banda
  - Semelhante a solicitações de atraso
  - RREQ pode incluir ambos os tipos
  - Usado durante o processo de descoberta de rota



QoS para AODV – Perda de QoS

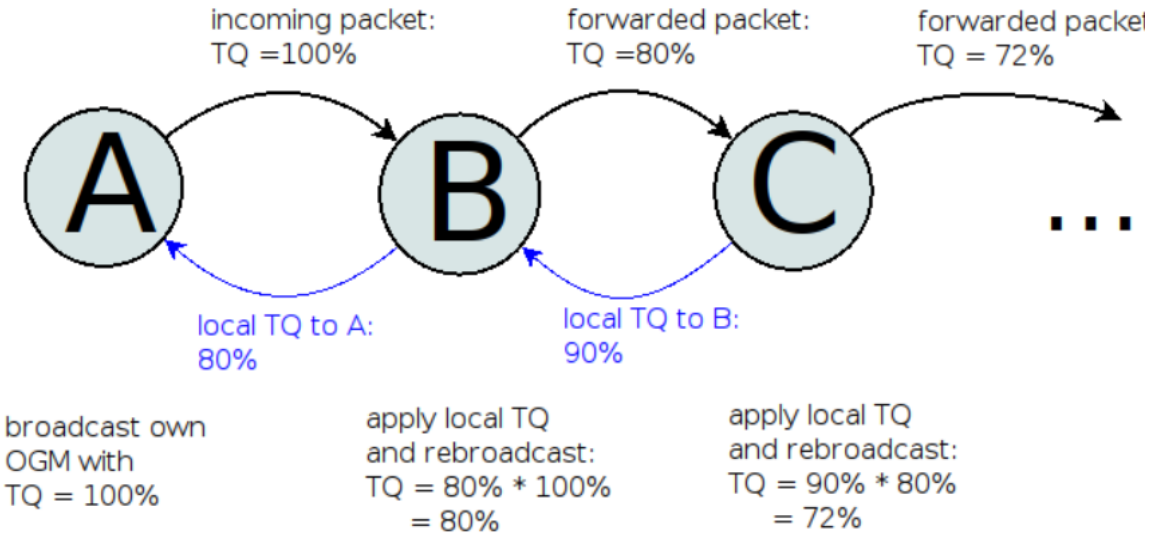
- Perdendo parâmetros de QoS
  - Se, após o estabelecimento, um nó detectar que a QoS não pode mais ser mantida, ele origina uma mensagem ICMP QoS_LOST para todos os nós dependentes.
    - Razão para manter uma lista de fontes solicitando garantias de atraso/largura de banda
- Razões para perder parâmetros de QoS
  - Aumento da carga de um nó
  - Por que um nó assumiria mais tarefas do que pode realizar?



TQualidade de transmissão (Batman v.3)

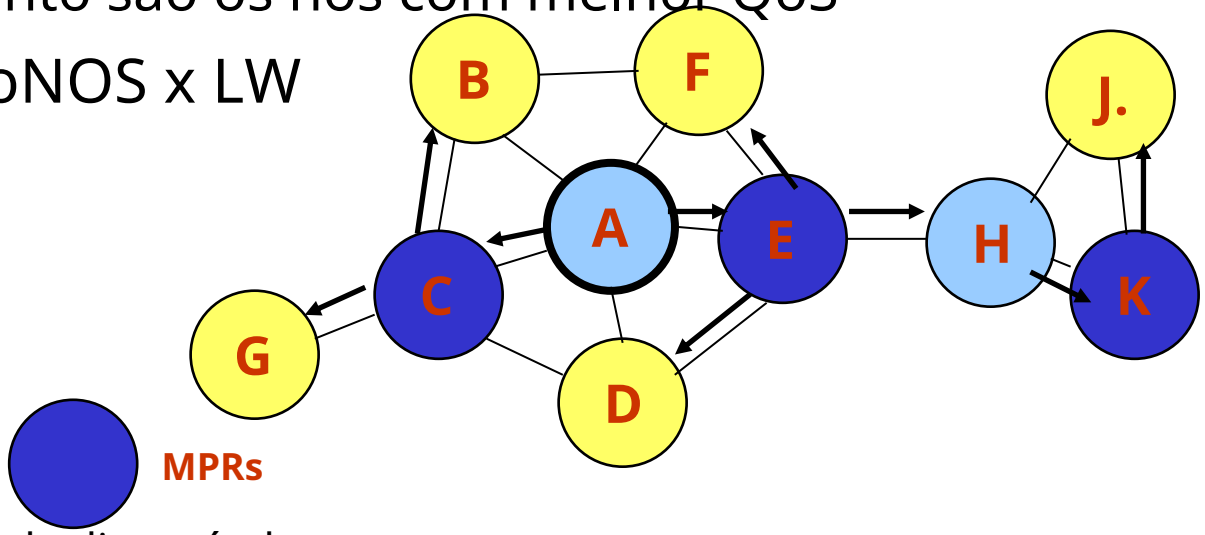
- Para adicionar a qualidade do link local no valor TQ é realizado o seguinte cálculo:
- TQ = TQ_{entrada} * TQ_{local}
- Exemplo: O nó A transmite o pacote com TQ máx. O nó B recebe, aplica o cálculo TQ e retransmite.

  Quando o nó C recebe o pacote ele sabe sobre a qualidade de transmissão para o nó A.



QoS-OLSR

- Relés multiponto são os nós com melhor QoS
- QoS=AB x PoNOS x LW



- AB: largura de banda disponível
- PoNOS: Porcentagem de Vizinhos na Outra Rua, apresenta o nível de diversidade dos vizinhos.
- LW: Lane Weight é usado para favorecer a seleção de MPRs de faixas que transportam a maior parte do fluxo de tráfego para aumentar sua estabilidade.

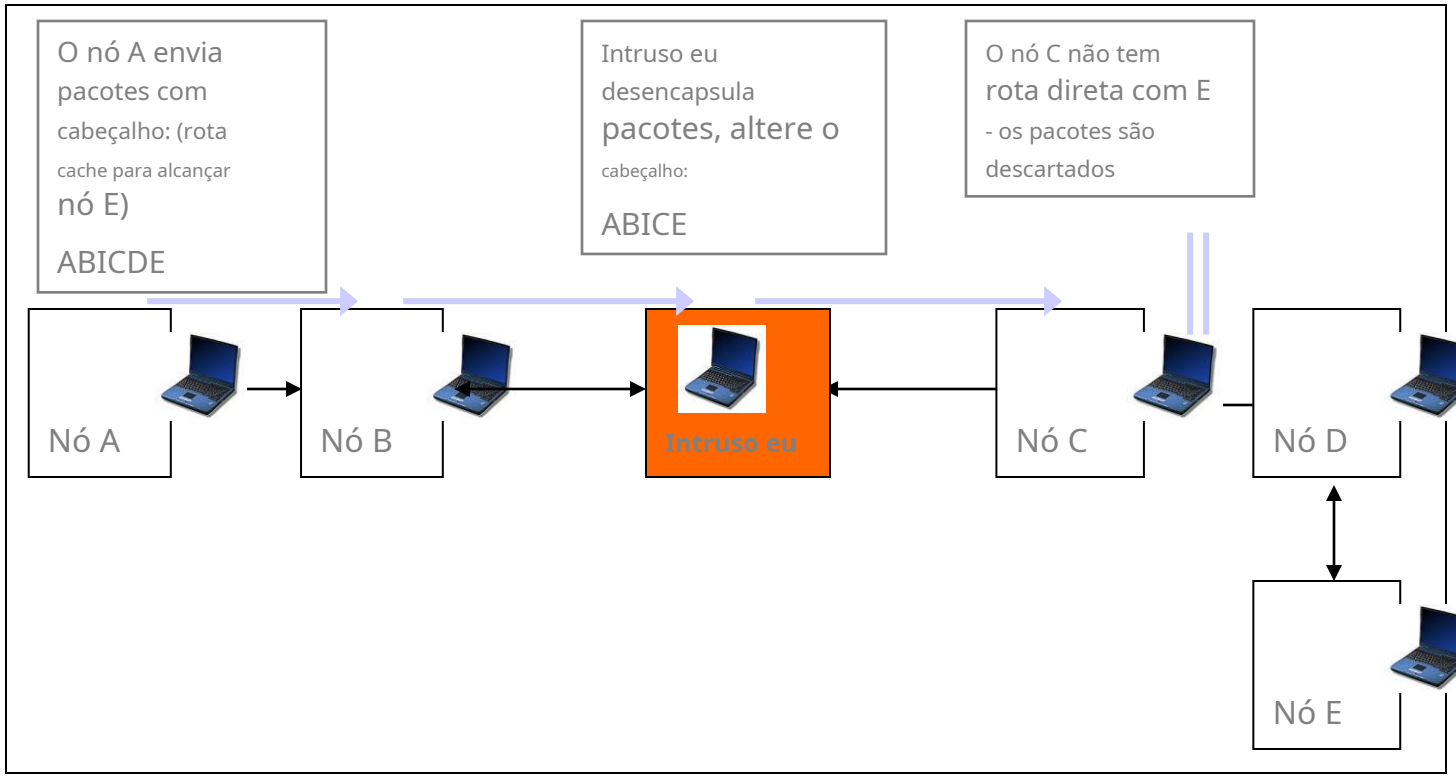
Segurança

Possíveis ataques de roteamento: ataques usando modificação

- Nó malicioso anuncia rotas melhores que os outros nós para ser inserido na rede
  - – Redirecionamento alterando o número de sequência da rota
  - – Redirecionamento com contagem de saltos modificada
  - – Ataques de negação de serviço (DOS) com rotas de origem modificadas
    - Um nó malicioso é inserido na rede através de uma das técnicas anteriores
    - O nó malicioso altera os cabeçalhos dos pacotes que recebe
    - Os pacotes não chegarão ao destino
    - A transmissão é abortada

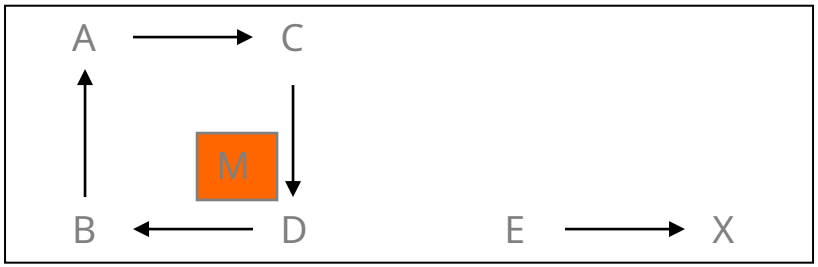
Possíveis ataques de roteamento: ataques usando modificação

- Ataques DOS com rotas de origem modificadas



Possíveis ataques de roteamento: ataques usando representação

- Usurpação da identidade de outro nó para realizar alterações
  - Falsificação de endereço MAC de outros nós
  - Formação de loops falsificando endereço MAC
    - Um nó malicioso M pode escutar todos os nós
    - Ele muda seu endereço MAC para o endereço MAC de outro nó
    - Anuncia a vários nós um caminho mais curto para chegar a X



    - X agora está inacessível por causa do loop formado

Possíveis ataques de roteamento: ataques usando fabricação

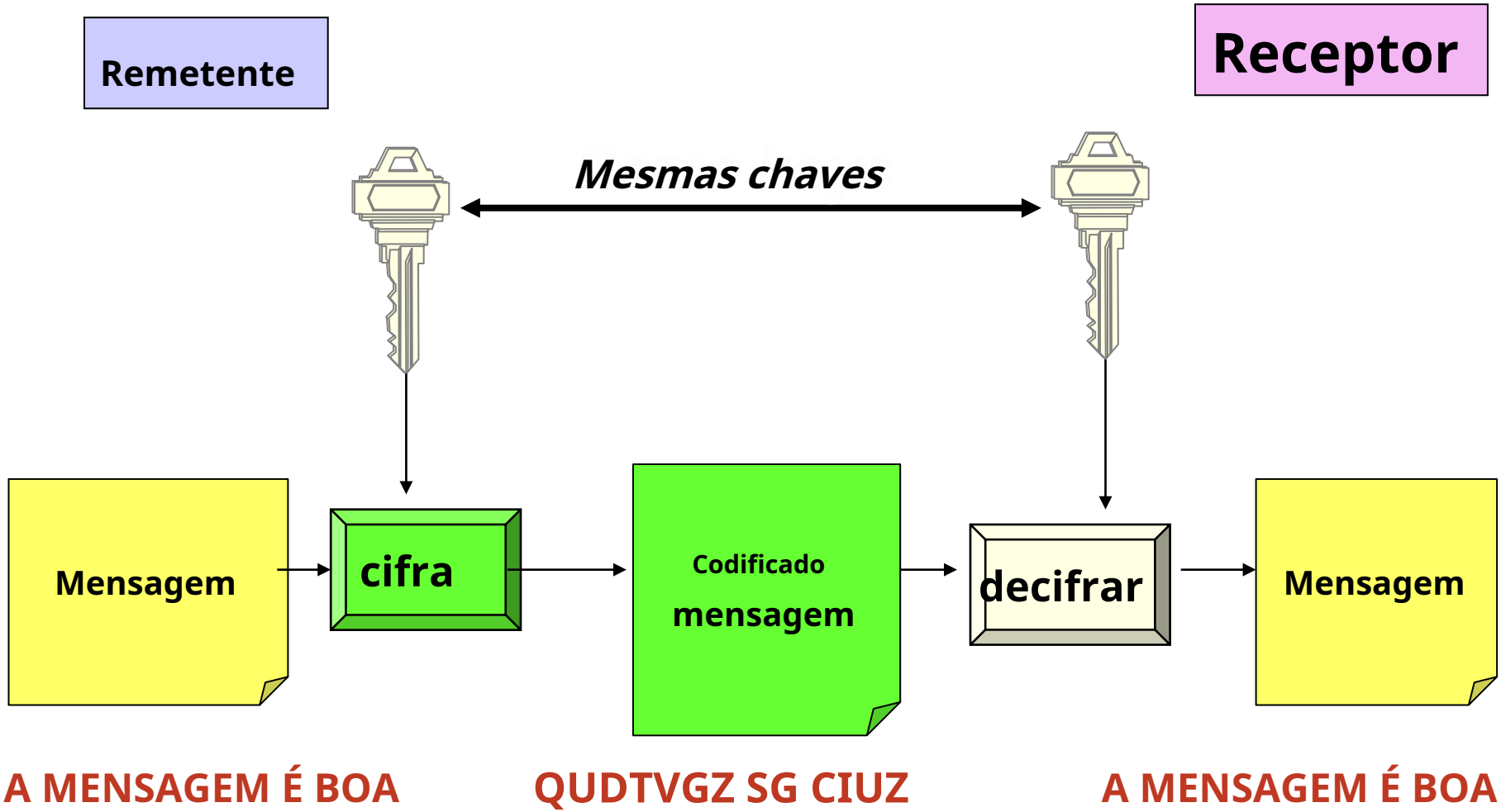
- Gera tráfego para perturbar o bom funcionamento de uma rede ad-hoc
  - Falsificação de mensagens de erro de rota
    - Isolar nós
  - Corrompendo o estado de roteamento
    - O hacker pode facilmente transmitir uma mensagem com um endereço IP falsificado, de modo que os outros nós adicionem esta nova rota para alcançar um nó especial S
    - O nó malicioso receberá os pacotes destinados ao S
  - Ataque de estouro de tabela de roteamento
    - O hacker pode enviar muitas rotas na rede para nós inexistentes até sobrecarregar o protocolo
  - Ataque de repetição
    - Hacker envia anúncios antigos para um nó
  - Ataque de buraco negro
    - O hacker anuncia uma rota de métrica zero para todos os destinos
    - Todos os nós ao seu redor encaminharão pacotes para ele

Gerenciamento de Chaves – noções básicas

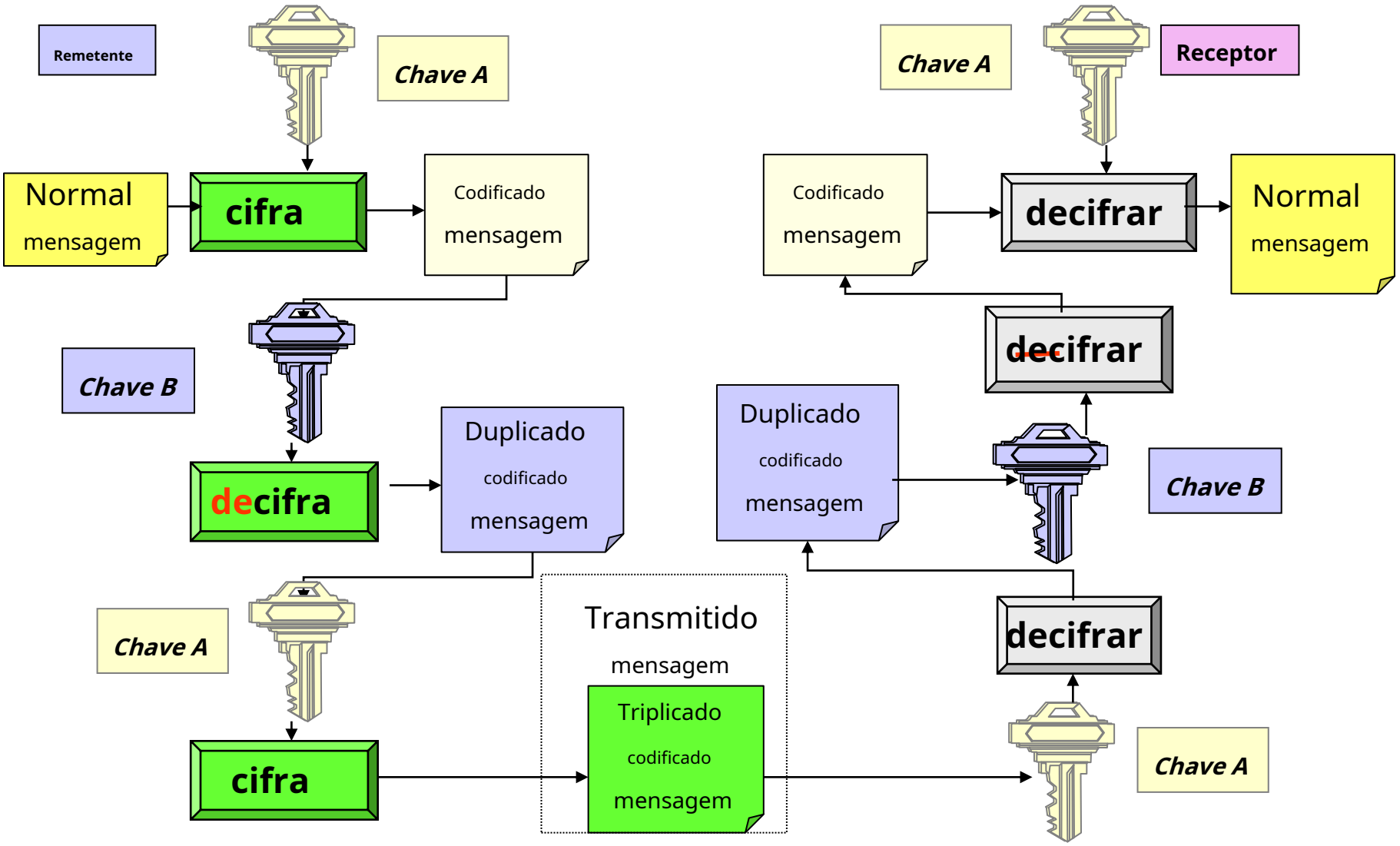
Cifra simétrica

- Vantagens
  - Rápido e relativamente seguro
    - Fornece integridade e privacidade
  - Maior comprimento de chave proporciona maior segurança
- Desvantagens
  - Requer o compartilhamento de uma chave secreta
    - Como?
  - Administração complexa e não escalável
    - É necessário distribuir as chaves
    - Uma chave para cada receptor

Cifra simétrica



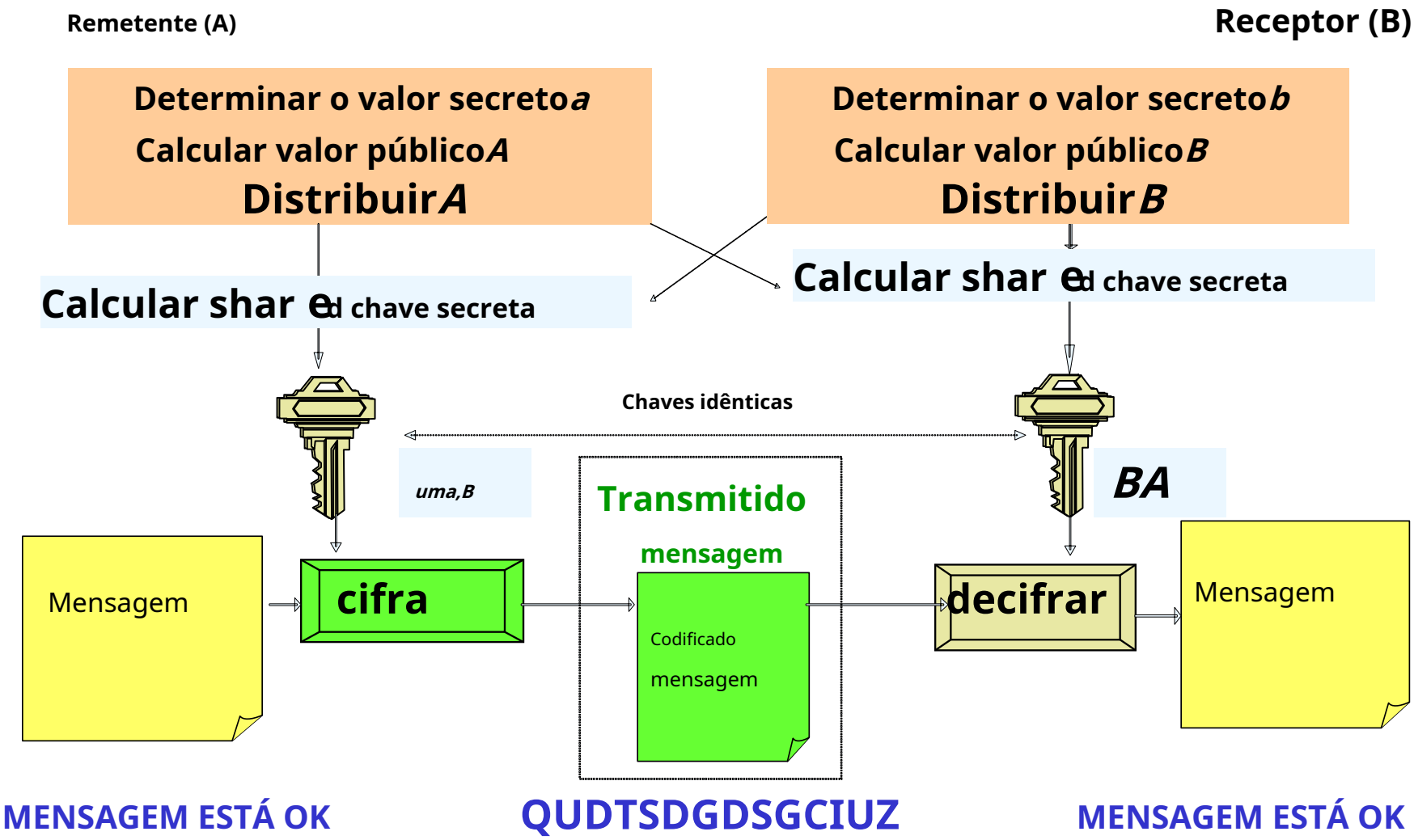
Mecanismos simétricos triplos (por exemplo, “3-DES”)



Cifra assimétrica

- Também conhecida como PKE - criptografia de chave pública
- Vantagens
  - Não é necessário compartilhar chaves secretas *a priori*
  - É escalável e versátil
- Desvantagens
  - Geralmente computacionalmente intensivo
  - Pode exigir uma autoridade de certificação
  - As chaves privadas devem ser confidenciais

Diffie-Hellman

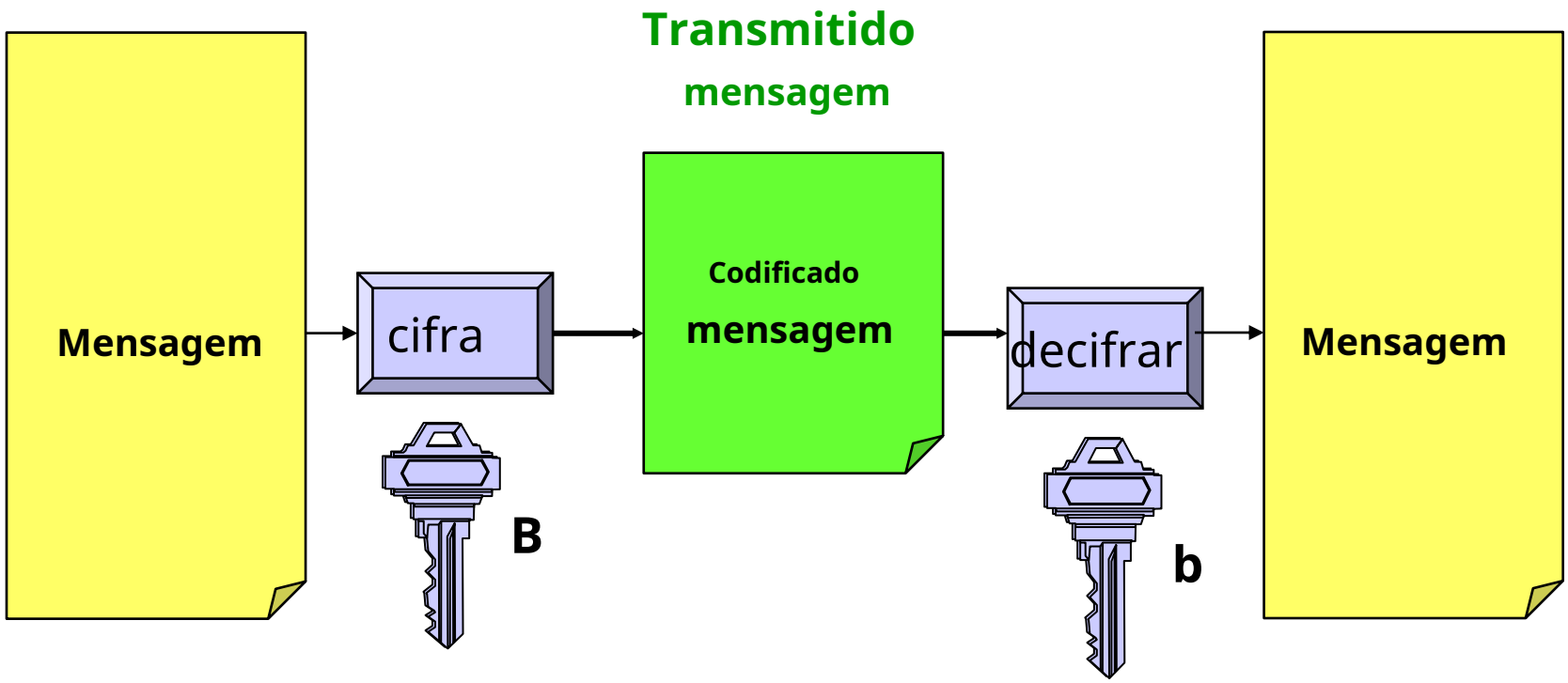


https://www.techtarget.com/searchsecurity/definition/Diffie-Hellman-key-exchange

Par público-privado para confidencialidade

Remetente (A,a)

Receptor (B,b)

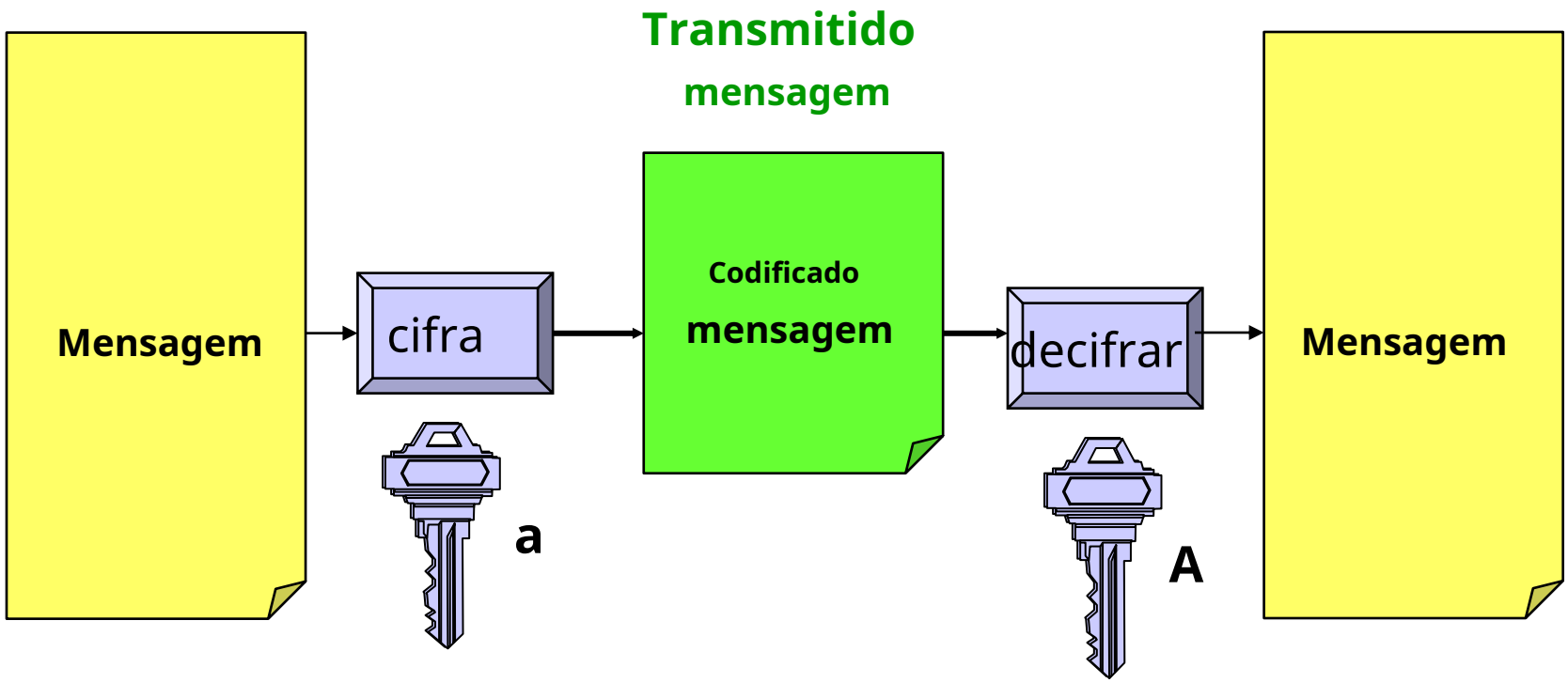


Garante confidencialidade sem garantia do remetente

Par público-privado para autenticação

Remetente (A,a)

Receptor (B,b)

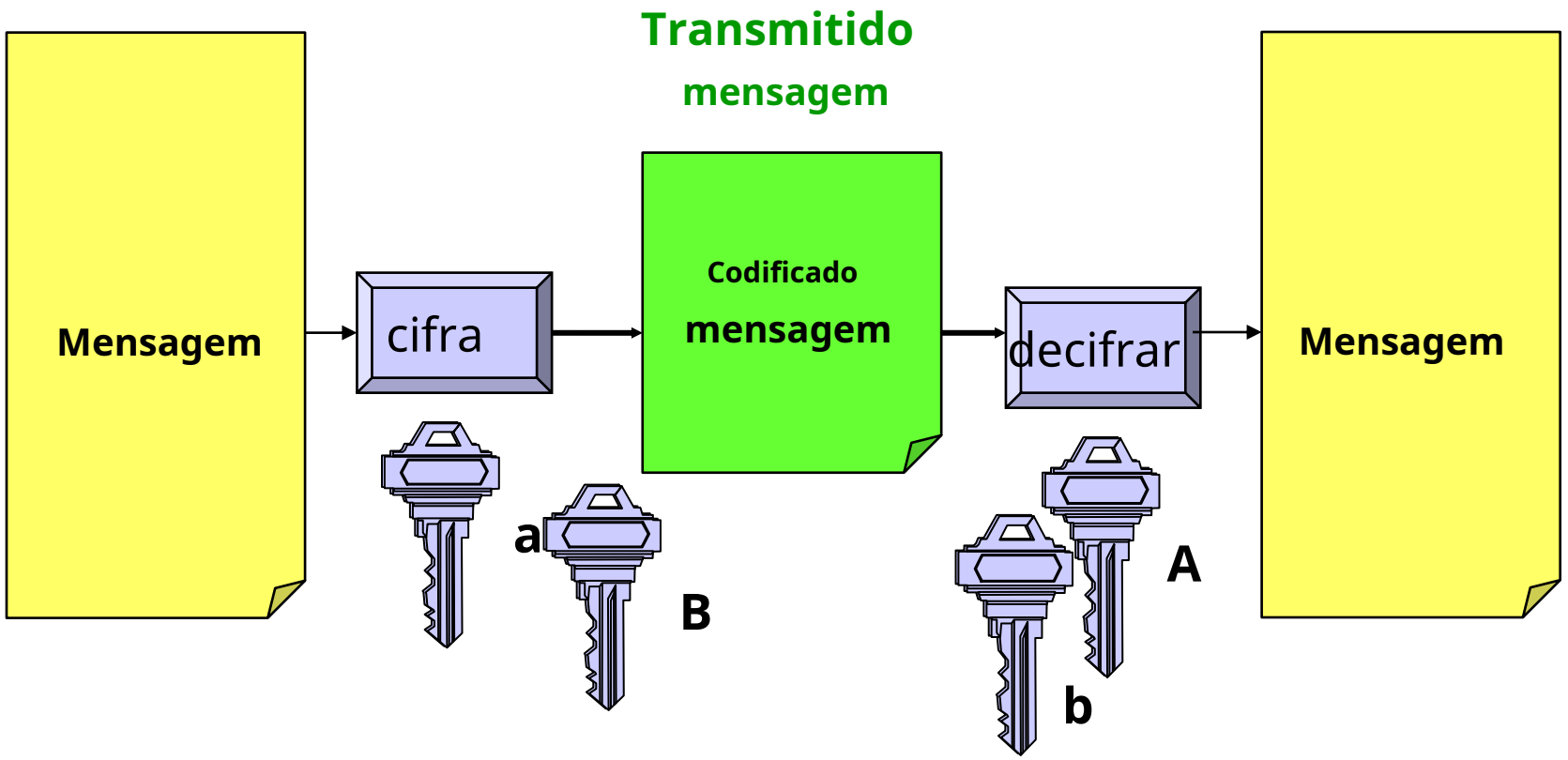


Autentica o remetente, pois somente o remetente terá a chave secreta a, correspondente à chave pública A

Par público-privado para confidencialidade e autenticação

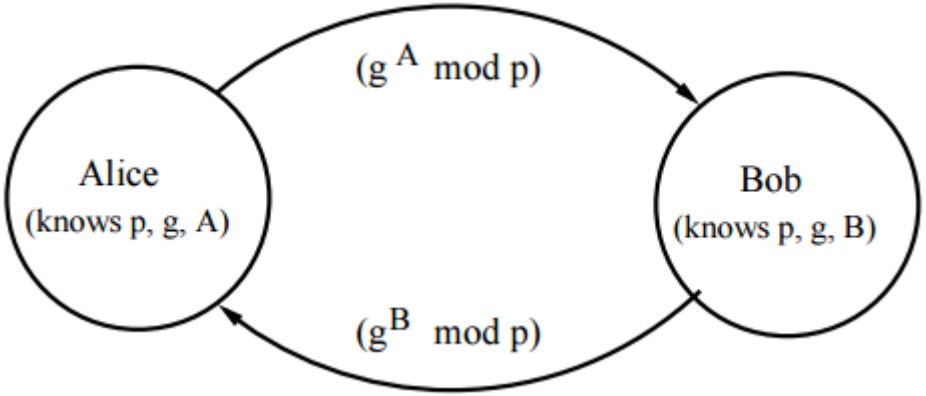
Remetente (A,a)

Receptor (B,b)



Garante a confidencialidade e autentica o remetente, pois somente o remetente terá a chave secreta a, correspondente à chave pública A

Diffie-Hellman



- Alice e Bob concordam com um número primo**p**e uma base**g**.
- Alice escolhe o número secreto**a**e envia para Bob (**g**$_{a}$moda**p**).
- Bob escolhe o número secreto**b**, e envia para Alice (**g**$_{b}$moda**p**).
- Alice calcula ((**g**$_{b}$moda**p**)$_{a}$moda**p**).
- Bob calcula ((**g**$_{a}$moda**p**)$_{b}$moda**p**).

- Alice e Bob usam esse valor como chave de sessão.**p**e**g**não precisa ser protegido.

Exemplo Diffie-Hellman

- Alice e Bob escolhem p = 23, por exemplo = 5.
- Alice escolhe a = 6 e envia 5^6 módulo 23 = 8.
- Bob escolhe b = 15 e envia 5^{15} módulo 23 = 19.
- Alice calcula 19^6 módulo 23 = 2.
- Bob calcula 8^{15} módulo 23 = 2.

- 2 é a chave compartilhada.

RSA (Rivest-Shamir-Adleman**) Chave**

- Cada usuário gera o par chave pública/privada
- Gera aleatoriamente 2 números primos grandes -**p**,**q**
- Calcula o módulo do sistema**N**=**p**.**q**
  - ø(N)=(p-1)(q-1)
- Seleciona uma chave aleatória**e**
  - onde 1<**e**<ø(**N**), mdc(**e**,ø(**N**))=1 mdc = máximo divisor comum
- Resolve a equação para encontrar a chave e decifra**d**
  - **e**.**d**=1 módulo ø(**N**) e 0≤**d**≤**N**
- Publica a chave pública: KU={**e**,**N**}
- Mantém a chave privada para decifrar: KR={**d**,**p**,**q**}

https://www.comparitech.com/blog/information-security/rsa-encryption/

Exemplo de RSA

1. Seleciona números primos: p=17 e q=11
2. Calcula $n=pq$=17×11=187
3. Calcula ø(n)=(p–1)(q-1)=16×10=160
4. Seleciona e: mdc(**e**,160)=1; escolhe e=7
5. Determina d: de =1 mod 160 e d<160: o valor é d=23 porque 23×7=161= 160+1
6. Publica a chave pública KU= {**e**,**N**}={7.187}
7. Mantém a chave secreta KR={**d**,**p**,**q**} = {23,17,11}

Uso de RSA

- Para cifrar a mensagem M, o remetente:
  - Obtém uma chave pública do receptorKU={e,N}
  - Calcula:C=M^emódulo N,onde0≤M<N
- Para decifrar C:
  - Usa sua chave privadaKR={d,p,q}
  - Calcula:M=C^dmod N
- A mensagem M deve ser menor que o módulo N

- Mensagem M = 88 (88<N=187)
- Cifra: C = 88^7módulo 187 = 11; e=7
- Decifrar: M = 11^{23}módulo 187 = 88; d=23

RSA Real

p

12131072439211271897323671531612440428472427633701410925634549312301964
37304208561932419736532241686654101705736136521417171171379797429933487
1062829803541

q

12027524255478748885956220793734512128733387803682075433653899983955179
85098879789986914690080913161115334681705083209602216014636634639181247
0987105415233

n

14590676800758332323018693934907063529240187237535716439958187101987343
87990053589383695714026701498021218180862924674228281570229220767469065
43401224889672472407926969987100581290103199317858753663710862357656510
50788371429711563734278891146353510271203276516651841172685983798867211
18372050855263466187400 53

$\phi(n)$

14590676800758332323018693934907063529240187237535716439958187101987343
87990053589383695714026701498021218180862924674228281570229220767469065
43401224889648313811232279966317301397777852365301547848273478871297222
05858745715289160645926971811926897116355507080264399952954964411681194
75165139381842966835212 80

e - the public key

$$65537$$

d - the private key

894894250092744443682285459217730939196695860658842574454978544564876748396298183909349419732628796167979706089172836798754993315741611138540888132754881105882471930775825272784379065040156806234235500672400424666656542323835029222154936232894721388664458187891279461234078077257026266440910365025372545139713

Encryption/Decryption

Encryption: $1976620216402300889624482718775150^{e} \bmod n$

350521113386730266902124239370533285118807608115799816206428023466858106231098502359430490809733862411137840407947041939782153784997654130836464387847409523069325349451950801838615742252262188798272324539128205968864403775360824656817500744174591514854074458625110234722355608230534977915189288202722577877 86

Decryption:

350521113386730266902124239370533285118807608115799816206428023466858106231098502359430490809733862411137840407947041939782153784997654130836464387847409523069325349451950801838615742252262188798272324539128205968864403775360824656817500744174591514854074458625110234722355608230534977915189288202722577877 86$^{d} \bmod n$

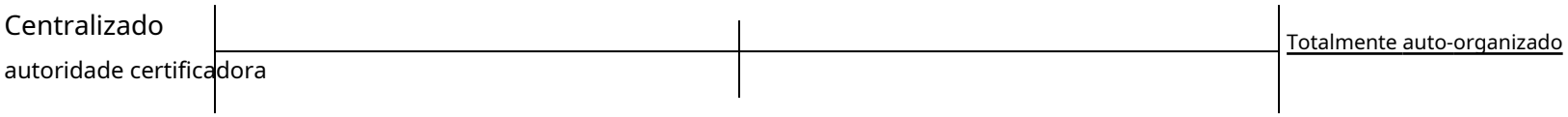
1976620216402300889624482718775150 (which is our plaintext)

Gerenciamento de chaves em redes ad-hoc

- Desafios
  - Nós móveis vulneráveis
    - Mais exposto a ataques físicos
  - Topologia de rede instável induzida por mobilidade
    - Mudança rápida na conectividade
    - Potencial partição de rede
  - Nenhuma infraestrutura para enviar informações importantes aos nós

Gerenciamento auto-organizado de chave pública (SOPKM)

- O escopo do gerenciamento de chaves



- Sistema de gerenciamento de chave pública
  - Como obter a chave pública autêntica de um usuário?
- Os usuários emitem certificados com base em conhecimentos pessoais
  - Certificado: ligação entre o nó e sua chave pública: chave pública, identidade e assinatura do emissor
  - Certificados armazenados e distribuídos pelos próprios usuários
- Repositório de certificados atualizado
- Repositório de certificados não atualizado
  - Certificados expirados

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1195151

SOPKM: Chaves Públicas e Certificados de chave pública

- Se um usuário *você* acredita que uma determinada chave pública K_v pertence a um determinado usuário *v*, então *você* pode emitir um certificado de chave pública no qual K_v é obrigado a *v* pela assinatura de *você*.
- Os certificados são emitidos com um período de validade limitado T_v
- Cada certificado contém seus prazos de emissão e expiração.

- *você* obtém a chave pública K_v de *v* através de um canal lateral
- *você* cria certificado para *v* para vincular o K_v e *v*
- *você* armazena o certificado de *v* em seu repositório local $G_{você}$ e envia o certificado para *v*
- Cada vez que um usuário *você* emite um certificado que vincula outro usuário *v* para sua chave pública K_v, *você* envia o certificado para *v*
  - *Desta forma, cada certificado é armazenado pelo menos duas vezes: pelo seu emissor e pelo utilizador a quem é emitido.*

SPKM: Atualizar repositórios

de gráficos de certificados

Certificado de

você para *v*, usando

você chave privada

pr $K_{você}(v, K_v)$

$G_{você}$ Atualizada

global

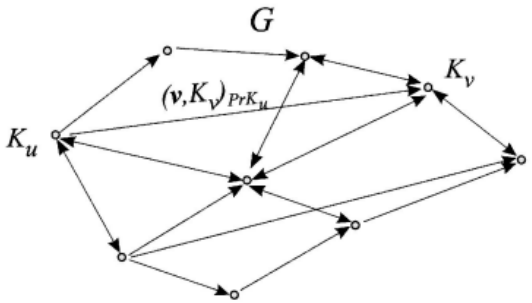
certificado de *você*

$G^N_{você}$ não atualizado

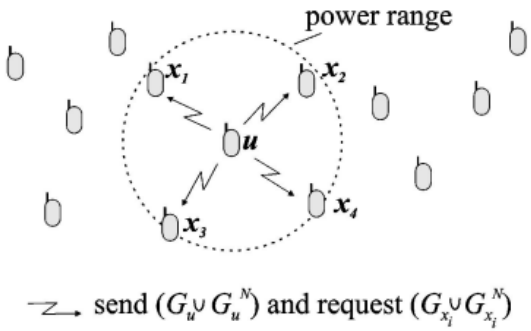
global

certificado de *você*

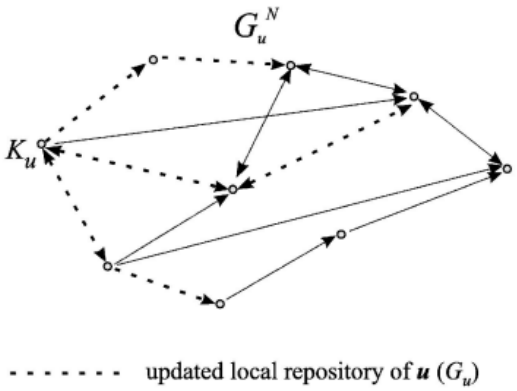
Step 0. Each user creates her own public/private key pair *K/PrK*



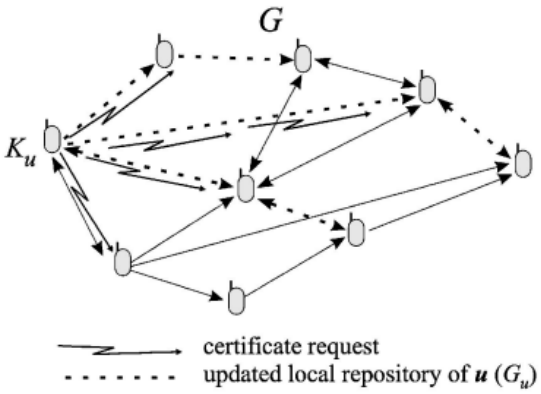
Step 1. Issuing of public-key certificates (creation of the certificate graph *G*)



Step 2. Certificate exchange



Step 3a. Node *u* constructs its updated repository from G_u^N



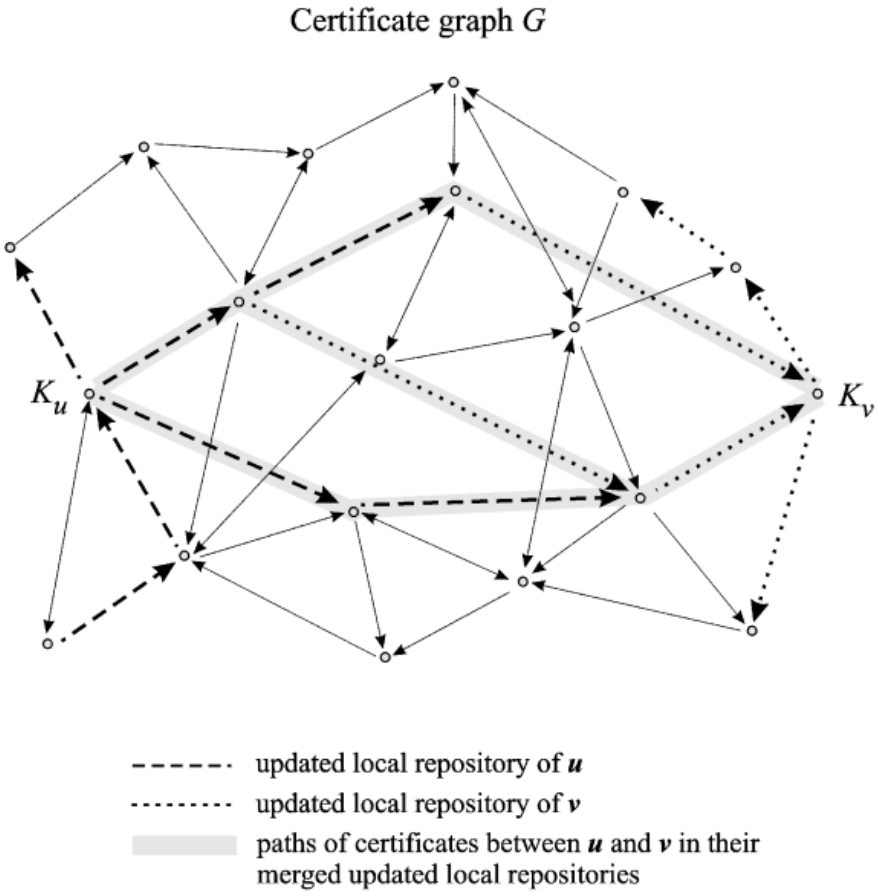
Step 3b. Node *u* constructs its updated repository by communicating with other nodes

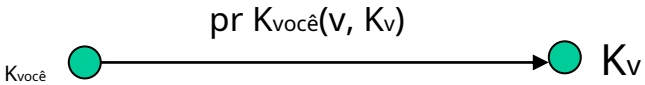
SOPKM: troca de certificados

- Cada nó *você* multicast seus subgráficos $G_{você}$ e $G_{N_{você}}$ para o seu vizinhos físicos (apenas 1 salto de distância)
- Nesta mensagem, *você* não envia certificados reais, mas apenas identificadores exclusivos apropriados (por exemplo, seus valores hash)
- Os vizinhos do nó *você* que recebem a mensagem de *você* responda com os valores hash dos certificados em seus repositórios atualizados e não atualizados.
- Nó *você* em seguida, verifica os valores recebidos com os certificados que possui e solicita de seus vizinhos apenas os certificados que não possui.

SOPKM: Conectividade Global Gráfico

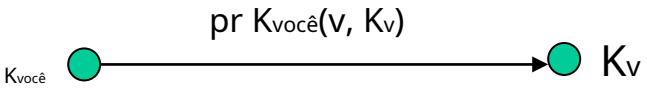
- Gráfico de certificado global
- Conectividade do gráfico do certificado
  - Fenômeno mundial pequeno
    - Devido às relações sociais
    - A mobilidade aumenta a conectividade
      - Nós/usuários móveis
        trocar seu público
        chaves sempre que se encontram
- Vértices: chaves públicas de alguns nós
- Edges: certificados de chaves públicas emitidos pelos usuários



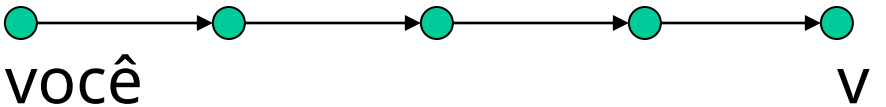


SOPKM: gráfico de certificado

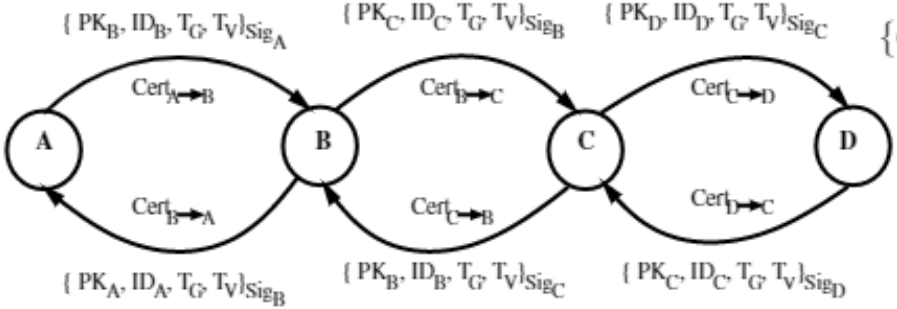
- Vértices: chaves públicas dos nós
- Edges: certificados de chaves públicas emitidos pelos usuários



- O usuário u deseja obter a chave pública do usuário v
  - Encontre uma cadeia de certificados de chaves públicas válidos que levem a v
    - O primeiro salto usa borda de você (certificado emitido por você)
    - O último salto é um certificado emitido por v
    - Nós intermediários confiáveis por meio do certificado anterior no caminho



$\{Cert_{A\to B}, Cert_{B\to C}, Cert_{C\to D}\}$



$\{Cert_{D\to C}, Cert_{C\to B}, Cert_{B\to A}\}$

SOPKM: atualização de certificado

- Antes de um certificado expirar, seu emissor emite uma versão atualizada do mesmo certificado, que contém um prazo de validade estendido
  - Versão atualizada é a atualização do certificado
- Cada nó emite periodicamente atualizações de certificado
  - Seu proprietário considera que as ligações de chave de usuário contidas nesses certificados estão corretas
  - Nenhuma informação sobre mau comportamento
- Um nó é capaz de estabelecer comunicação com qualquer emissor de certificado em algum momento dentro do período de validade do certificado
  - As atualizações de certificados serão trocadas regularmente
- Simples, mas precisa de sincronização de horário dos nós e decisão adequada do tempo de expiração (normalmente vários dias)

SOPKM: Certificado Revogação

- Cada usuário pode revogar um certificado emitido se
  - Acredita que a ligação da chave do usuário expressa nesse certificado não é mais válida
  - Sua própria chave privada está comprometida, revogue a chave pública correspondente
- Revogação de certificado
  - Explicitamente (revogar um certificado que tenha emitido)
    - O usuário emite uma declaração de revogação explícita
    - Cada nó possui uma lista de nós que solicitam atualizações para os certificados emitidos
    - Não é necessário enviar a revogação para todos os nós, mas apenas para os nós que atualiza regularmente.
    - A revogação do certificado alcançará outros nós, com atraso no tempo de convergência da troca de certificados.
  - Implicitamente
    - Prazo de validade do certificado
    - Dentro deste período, os nós poderão atualizar seus repositórios de certificados
      - Se dentro de um determinado período um nó não for capaz de atualizar alguns dos certificados em seu repositório local atualizado, o nó poderá reconstruí-lo usando apenas os certificados disponíveis para atualização.

SOPKM: usuários maliciosos

- O mecanismo de troca de certificados permite que os nós reúnam praticamente todos os certificados de *G*
  - Os nós verificam as ligações de chave do usuário nos certificados que possuem e detectam quaisquer inconsistências (ou seja, certificados conflitantes).
  - Se contiverem ligações de chave de usuário inconsistentes (ou seja, se ambos os certificados contiverem o mesmo nome de usuário, mas chaves públicas diferentes)
  - Se contiverem a mesma chave pública, mas estiverem vinculados a nomes de usuário diferentes.
- Várias trocas de certificados para resolver conflitos, se necessário

Sem fio ad-hoc autoprotegido redes (SSAWN)

- Normalmente, uma entidade só é confiável se for verificada por uma autoridade central, o que não pode ser o caso em redes sem fio e ad-hoc
- Objetivo de uma rede autossegura
  - Obtenha alta garantia de segurança
  - Alta taxa de sucesso
  - Comunicação eficiente
- Modelo de confiança localizada: uma entidade é confiável, se houver *k* entidades confiáveis afirmam isso dentro de um determinado período de tempo
  - *k* entidades normalmente entre os vizinhos de um salto da entidade
    - Na prática, preocupa-se mais com a confiabilidade de seus vizinhos imediatos - um nó se comunicará com o resto do mundo através de seus vizinhos de um salto.
  - Uma vez que um nó é confiável pela sua comunidade local, ele é globalmente aceito como um nó confiável.
  - Caso contrário, uma entidade localmente desconfiada será considerada indigna de confiança em toda a rede.

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1021731

SSAWN: segredos compartilhados

- O mecanismo de criptografia usa chaves assimétricas RSA
- Chave Secreta Global (SK) e a Chave Pública (PK) correspondente
  - A funcionalidade SK é 'distribuída' entre os nós
  - Quaisquer K nós contendo um segredo parcial formam uma Autoridade de Certificação (CA) distribuída
- SK é usado para assinar certificados para todos os nós da rede.
- Um certificado assinado por SK pode ser verificado pela conhecida chave pública PK.
- Limite de compartilhamento secreto
  - Cada nó possui uma parte do segredo
    - ID exclusivo, derivado do endereço do nó
    - Mecanismo para detecção local de nós com mau comportamento
    - Pelo menos K nós vizinhos de um salto
    - Par de chaves para cada nó (chaves públicas e secretas)

SSAWN: Operação Básica

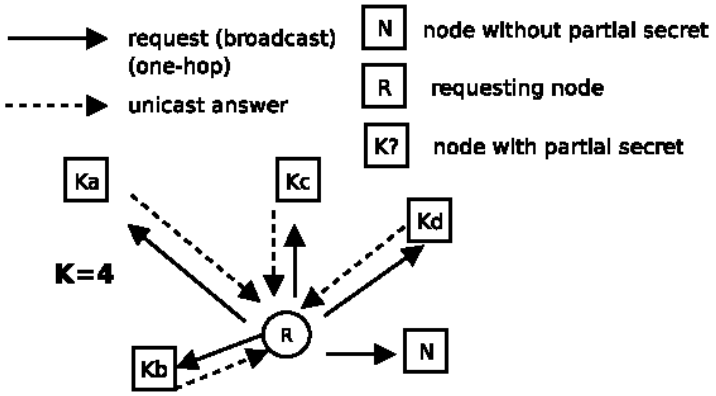
- Operação basica
  - PKI distribuída
    - A chave privada do sistema é dividida em nós do servidor
    - Quórum de *k* servidores produzem atualização de certificado
    - Estrutura do certificado

ID_i	T_{valid}	K_i	*flag*	*ver*	*sign.*	*issuer*	*Algo.*

  - Opera em fases
    - Formação/manutenção de grupo de servidores
    - Atualização/revogação de certificado
    - Atualização/renovação de chave compartilhada
- SK não é visível, conhecido ou recuperável por nenhum nó da rede
- Cada nó possui um certificado assinado com SK.
- Presume-se que a PK seja bem conhecida para verificação de certificados.
- Nós sem certificados válidos têm acesso negado a quaisquer recursos de rede, como roteamento e encaminhamento de pacotes

SSAWN: segredos compartilhados

- Chave secreta parcial em função dos IDs dos nós
  - – Geração de um polinômio de ordem K-1, conhecido apenas na configuração inicial
  - – K nós que possuem um compartilhamento secreto parcial recuperam SK usando interpolação de Lagrange
  - – A coalizão de nós K-1 que possuem um compartilhamento secreto parcial não possui nenhuma informação sobre SK
- Nó que deseja usar a CA distribuída
  - – Entre em contato com K nós que possuem um compartilhamento secreto parcial
  - – K nós vizinhos de um salto
    - É mais fácil coletar informações confiáveis sobre o mau comportamento de nós mais próximos
  - – PK é conhecido por todos os nós



SSAWN: segredos compartilhados

- Após o recebimento de v_{eu} solicitação de certificação, um nó verifica seus registros.
  - Se seu registro mostrar v_{eu} como um nó legítimo com bom comportamento, ele retorna um certificado "parcial" aplicando sua parcela de SK.
  - Caso contrário, a solicitação será descartada.
- Ao coletar k certificados parciais, v_{eu} combina-os para gerar o novo certificado completo como se fosse de um servidor CA
  - Ao receber k certificados parciais da coalizão, nó v_{eu} multiplica-os entre si para recuperar seu certificado completo (interpolação de lagrange – polinômio que passa pelos diversos pontos, https://mathworld.wolfram.com/LagrangeInterpolatingPolynomial.html)
- Um nó com mau comportamento ou quebrado não poderá renovar seu certificado.
- Um certificado válido representa a confiança de uma coalizão de k nós
  - Os nós com certificados válidos são globalmente confiáveis.
  - Cada nó contribui para o gerenciamento e manutenção geral da confiança, monitorando e certificando seus nós vizinhos

Abordagens de reputação

Nós bem comportados?

- Explore a confiabilidade de cada nó
- Roteamento e comunicação através de nós de alta reputação
- Proteja o tráfego de rede contra nós com mau comportamento
- Minimizar a interação com nós que se comportam mal

- Avalie a reputação deste nó
- Isso precisa ser feito em uma abordagem distribuída por cada nó

  – Cada nó monitora o comportamento de cada nó vizinho

  – Os nós enviam suas informações para outros nós

Comportamento e reputação

- Explorar a forma como um nó lida com o encaminhamento de pacotes: encaminhar, alterar, injetar pacotes;
- Avaliar através da observação do comportamento de um nó
- Informações de reputação trocadas periodicamente entre vizinhos
- Nível de confiança: o nível de confiabilidade nos relatórios provenientes de um determinado nó
- As interfaces de rede dos nós precisam suportar o modo promíscuo
  - Se o nó A estiver dentro do alcance do nó B, ele poderá ouvir as comunicações de e para o nó B

Nós bons ou ruins?

- Sistema auto-organizado
  - Descentralizada: a cooperação pode ser imposta, mas não garantida
  - Reconhecer e sancionar comportamentos intoleráveis, recompensar membros obedientes
- Cada nó observa o comportamento e avalia a reputação de seus vizinhos
- A reputação pode ser usada para selecionar o caminho mais confiável e seguro
- Nós amigáveis, egoístas (não transmitem) e maliciosos (encaminham mal, injetam pacotes)
  - Se o comportamento mudar, a reputação do nó muda de acordo

Reputação

- A reputação total é uma combinação de
  - Reputação em primeira mão (vizinhos)
  - Reputação de segunda mão (informações de vizinhos)
- O comportamento de um nó segue uma distribuição
  - Número de pacotes observados de um nó
    - Número de pacotes não transmitidos ou alterados
  - Probabilidade de pacotes bem transmitidos (reputação em primeira mão) $>x$
    - Comportamento amigável (abordagem Bayesiana)
- Monitoramento colaborativo
  - Trocar reputação em primeira mão com vizinhos
  - Teste de desvio para detectar relatórios falsos
  - Probabilidade de bem transmitido x visto de diferentes nós $<$valor de desvio d
    - Relatórios confiáveis (abordagem Bayesiana)

Reputação

- Relatório confiável de A sobre B
  - *T = confiável*
  - *N = não confiável*
- Valor de confiança $\alpha = T/(T+N)$
- Mesclar informações de reputação de primeira mão e de segunda mão
- A reputação de segunda mão é descontada pelo fator confiança
- A reputação do nó B vista pelo nó A é a reputação de primeira mão mais o fator de confiança de uma reputação de segunda mão de B vista por C (e de outras reputações de segunda mão)
  - $R_{AB} = F_{AB} + \alpha F_{CB}$

Reputação no normal operação da rede

- Escolha de nós para formar um caminho da origem até um destino
  - Reputação dos nós no algoritmo de escolha
- Escolha de nós para ingressar em um gráfico de certificação
  - Nós com alta reputação
- Escolha de nós para obter as partes-chave ou pares de chaves
  - Nós com alta reputação

- A reputação de um nó muda com o tempo
- Os nós podem ficar fora da comunicação da rede por um período específico