VU Network Security Advanced Topics
Lecture 3

IPv6 Security

Tanja Zseby
TU Wien

WS 2018/19

institute of telecommunications

TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

Recap: IPv6

IPv6 Features

- Much larger address space
- Simple extensible header
- Hierarchical adresses to simplify aggregation
- Address autoconfiguration
- Extension headers
- Integrated multicast
- Integrated network layer security
- Mobility support
- QoS support
- Jumbogramms

IPv6 Header

+	0-3	4-7	8-11	12-15	16-23	24-31
0	Version	Traffic Class		Flow Label		
32	Payload Length				Next Header	Hop Limit
64	Source Address					
96						
128						
160						
192	Destination Address					
224						
256						
288						

IPv4 header:

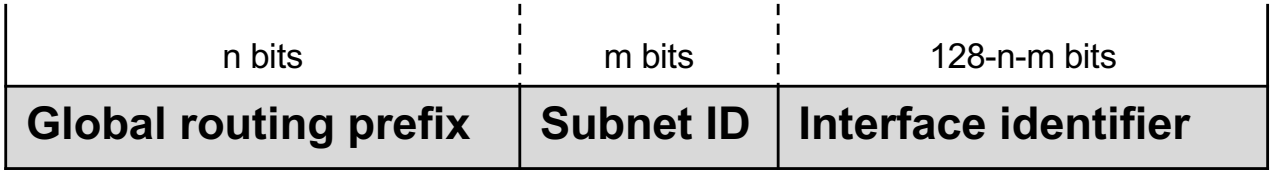
	0-3	4-7	8-11	12-15	16-18	19-23	24-27	28-31
0	Version	IHL	Type of Service		Total Length			
32	Identification				Flags	Fragment offset		
64	TTL		Protocol		Header Checksum			
96	Source Address							
128	Destination Address							
	Options							Padding

old
new
similar
removed

IPv6 Hierarchical Addresses

- Hierarchical addresses ➔ simple aggregation

IPv6 Global Unicast Address Structure (General Format)



- Global Routing Prefix
  - Typically hierarchically- structured value assigned to a site (cluster of subnets)
- Subnet ID
  - An identifier of a subnet within the site
- Interface Identifier (IID)
  - Unique identifier for an interface within a subnet
  - Usually 64 bit long

IPv6 Address Configuration Options

- Manual configuration
- ***Stateful*** Address Autoconfiguration: DHCPv6 [RFC3315, RFC4361]
  - Stateful ➔ DHCP server maintains list of assigned addresses
  - Provides addresses from DHCP server
  - Provides further configuration information
- ***Stateless*** Address Autoconfiguration (SLAAC) [RFC4862]
  - Stateless ➔ no list required
  - Hosts configure their addresses themself
- Stateless DHCPv6 [RFC 3736]
  - Subset of DHCPv6
  - provides no addresses, only additional configuration information (e.g., DNS servers, SIP servers, NTP-Server)
  - Works together with stateless address autoconfiguration or manual configuration

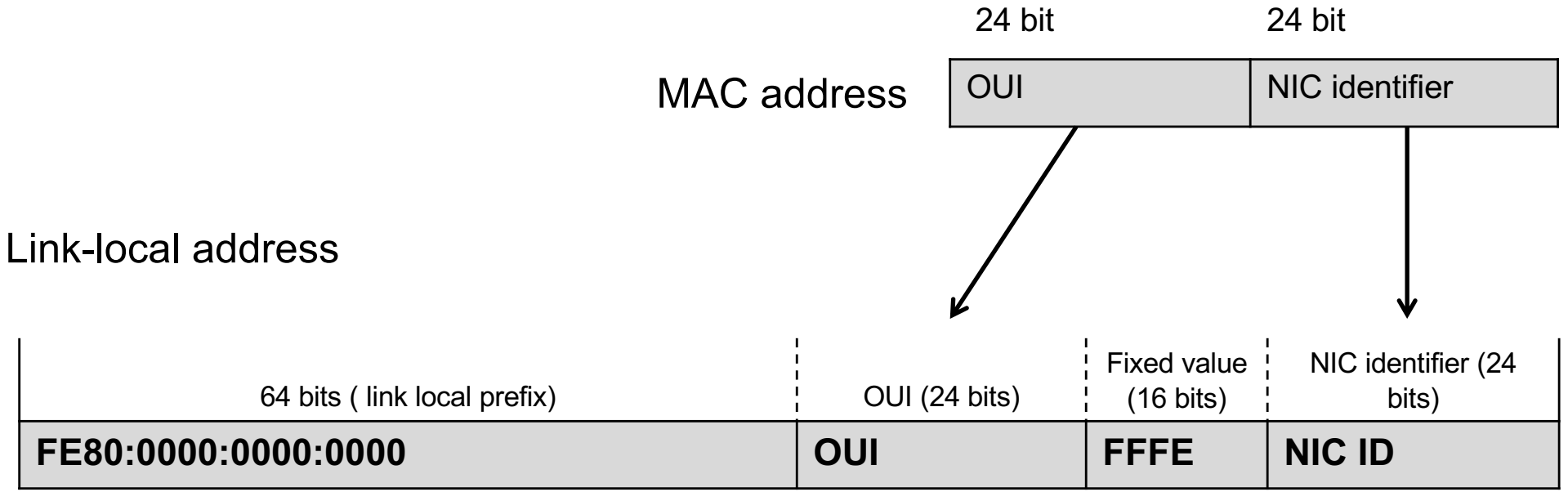
Stateless Address Autoconfiguration (SLAAC)

Defined in RFC 4862

1. Creation of link-local address
   - when interface initialized or attached to network
2. Check for address conflicts
   - Duplicate Address Detection (DAD)
3. Creation of global address
   - Adding routing prefix

1. Creation of Link-Local Address

- Host calculates link local address
  - First 64 bit: fe80:0000:0000:0000 (link local prefix)
  - Last 64 bit: Extended Unique Identifier (EUI) derived from MAC address



2. Duplicate Address Detection (DAD)

- Host checks if tentative link local address is unique
- Sends Neighbor Solicitation:
  - ask other hosts whether any has same address
- May receive Neighbor Advertisement
  - another host already uses address
- If conflict detected ➔ Manual address configuration

- Conflicts expect often?
- MAC address unique
  - ➔ Address collision very unlikely
  - exception: manipulated MAC addresses

3. Creation of Global Address

- Router sends ***Router Advertisements***
  - periodically to all-nodes multicast address
  - Contains: network prefix, lease timeout, Maximum Transfer Unit (MTU), hop count
- Host may send ***Router Solicitations***
  - to the all-routers multicast group
  - to request Router Advertisement and speed up process
- Host generates address from link local address and prefix
- Host performs Duplicate Address Detection (DAD) on global address (recommended)

64 bits	OUI (24 bits)	Fixed value (16 bits)	NIC identifier (24 bits)
Prefix (from router advertisement)	**OUI**	**FFFE**	**NIC ID**

IPv6 Security

Search for Victims in IPv6 Networks

- Scanning in IPv4
  - Address 32 bits, host part typically 8 bits
  - IPv4 (/24) ➔ 2^8 = 256 hosts
  - Scanning time (with 1 probe/second) for network: < 5 minute
- Scanning in IPv6:
  - Address 128 bits, host part typically 64 bits
  - ➔ 2^{64} addresses in network
  - Scanning time (with 1 probe/second) for network: > 500 billion years

- BUT: only true if addresses randomly assigned!

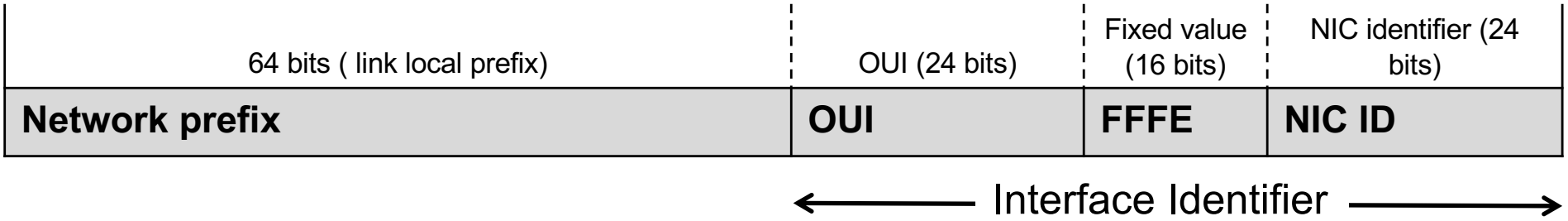
Search for Victims

When are addresses not random?

- Sequential numbering
  - [prefix]::1, [prefix]::2, [prefix]::3, ... ➔ easy to guess
- Simple to remember addresses
  - [prefix]::aaaa:ffff:ffff:eee, [prefix]::dead:beef, [prefix]::abad:1dea ➔ reduced search space
- Stateless Address Autoconfiguration (SLAAC)
  - Based on device ID ➔ reduction of search space

Stateless Autoconfiguration

IPv6 address after autoconfiguration:



- Possibilities for Interface Identifier
  - OUI – vendor-specific ➔ easy to guess
  - FFFE ➔ fixed, known
  - ➔ only unknown is NIC ID ➔ 24 bits
  - $2^{24} << 2^{64}$ ➔ Scanning time: 194 days

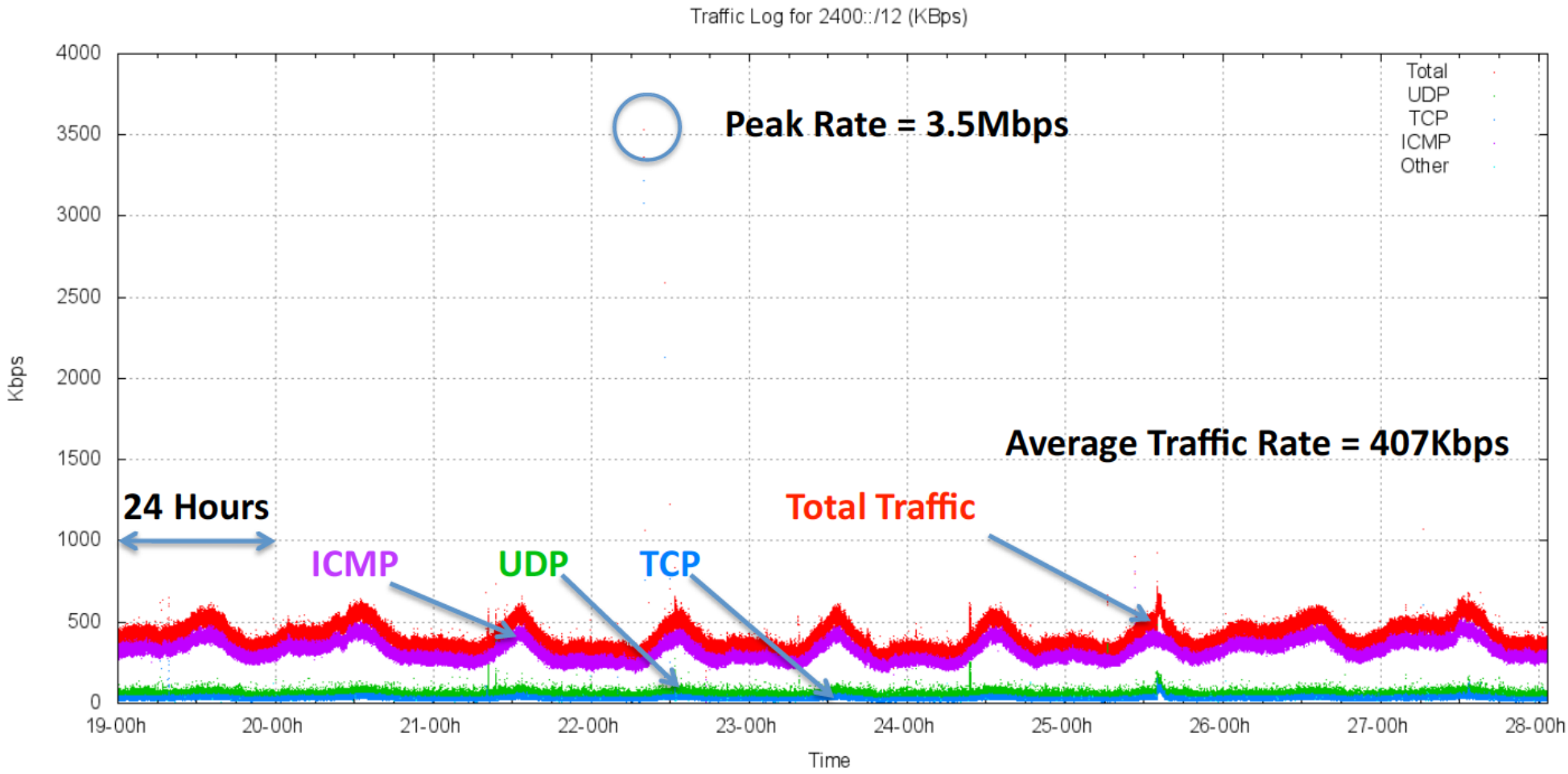
Search for Victims

- Other methods to find IPv6 hosts
  - Hosts with DNS entry
  - Log files (e.g. from website logs, email headers)
  - On-link: neighbor discovery, application traffic,…

- IPv6 Network Scanning ➔ see RFC 5157

IPv6 Darkspace?

Monitoring 2400::/12



Source: Geoff Huston http://labs.apnic.net/presentations/store/2010-11-18-ipv6-background-radiation.pdf

Secure Neighbor Discovery (SEND)

Neighbor Discovery Protocol (NDP) [RFC4861]

- Router Discovery (RD)
  - Hosts locates local routers on link
- Address Autoconfiguration
  - Configure an address automatically
- Address Resolution
  - Determine link-layer address of on-link destination
- Neighbor Unreachability Detection (NUD)
  - Check reachability of neighbors (hosts, routers)
- Duplicate Address Detection (DAD)
  - Check if address is already in use by another node
- Redirection
  - Router informs host about better first-hop node

Neighbor Discovery Protocol (NDP)

- Internet Control Message Protocol for IPv6 (ICMPv6)
  - Used for NDP and other functions
  - Multicast functions (membership)
  - NDP messages are ICMPv6 messages

- Uses IP multicast
  - ff02::1 all nodes on the local network segment
  - ff02::2 all routers on the local network segment

NDP Scope

- Protocol used in local network
  - Nodes connected on link layer

- Routers ***don't forward*** NDP messages
  - NDP messages cannot come from outside
  - Hop limit value is 255 ➔ detect if packet was routed

- NDP Attacks only in **local** network

Security Problems with Neighbor Discovery

- NDP assumes **trusted nodes** on link
  - Nodes trust each other to behave correctly
  - Do not send messages with false information

- But: for many environments this cannot be ensured
  - Malicious nodes may forge NDP message

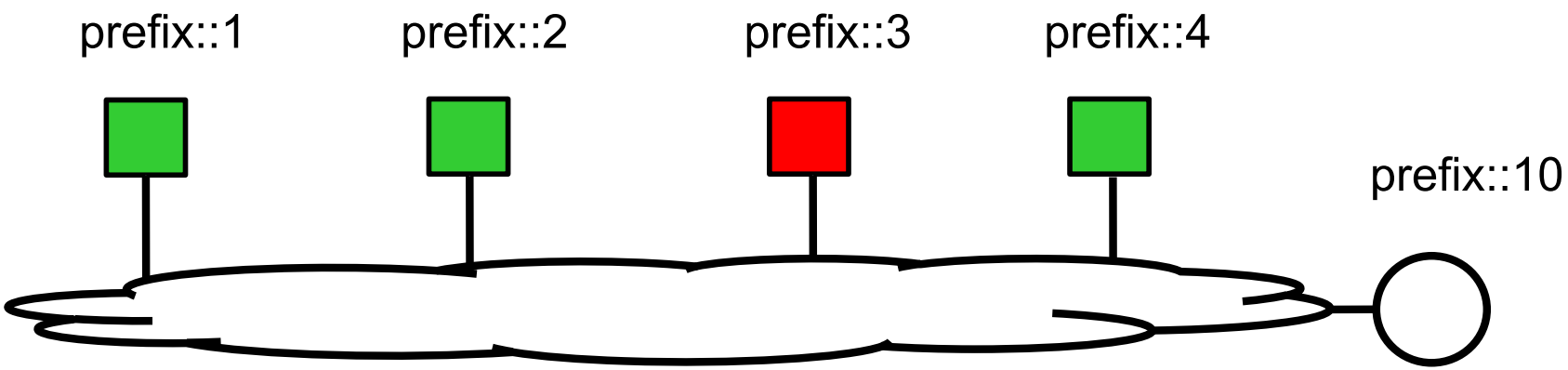
- ➔ RFC 3756: Trust Models and Threats

Trust Models [RFC 3756]

- All nodes trusted
  - Authenticated nodes trust each other
  - Nodes under one administration
  - Physically protected or secured on link layer
  - E.g. wired LAN with 802.1X access control
  - Example: Corporate Intranet
- Only routers trusted
  - link-layer authentication between nodes and router
  - Example: public WiFi
- No node trusted
  - No operator or trusted 3rd party
  - Example: ad hoc network

Neighbor Discovery Attacks

Malicious node in local network:

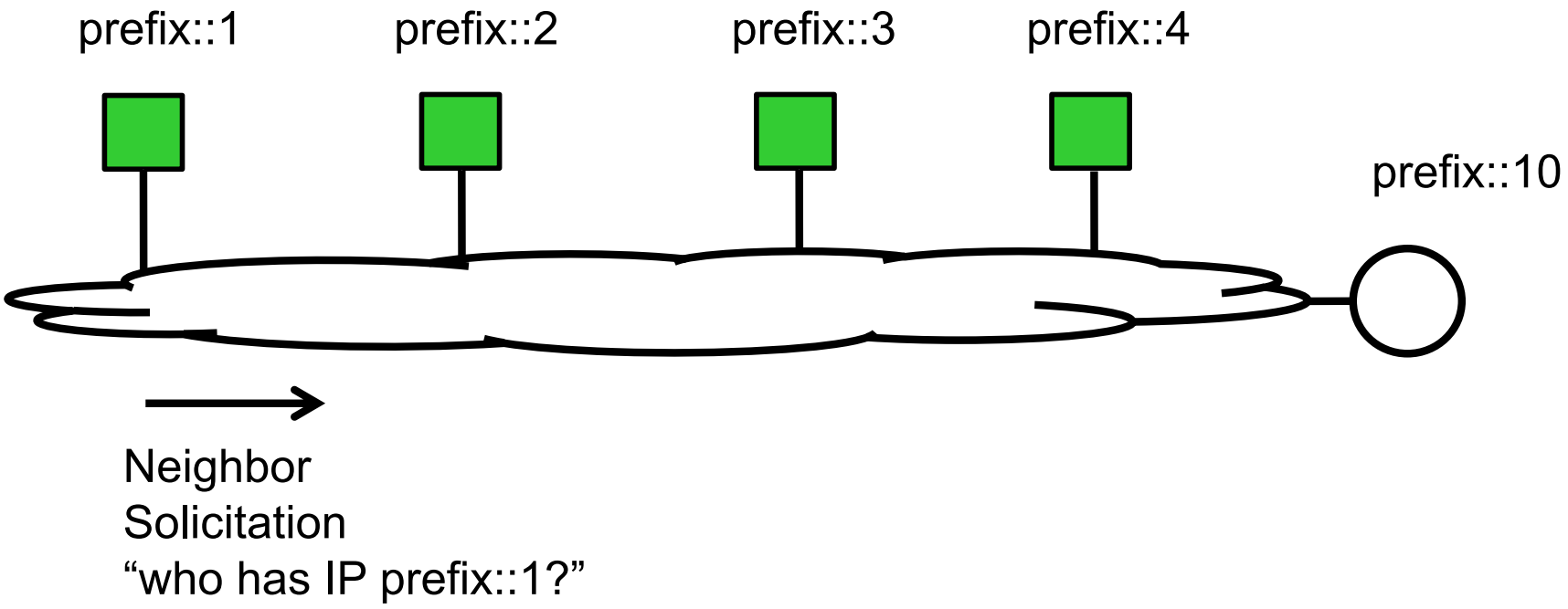


What can happen?

SLAAC Duplicate Address Detection

Purposes: check if own generated address can be used

- new node joins
- generates address

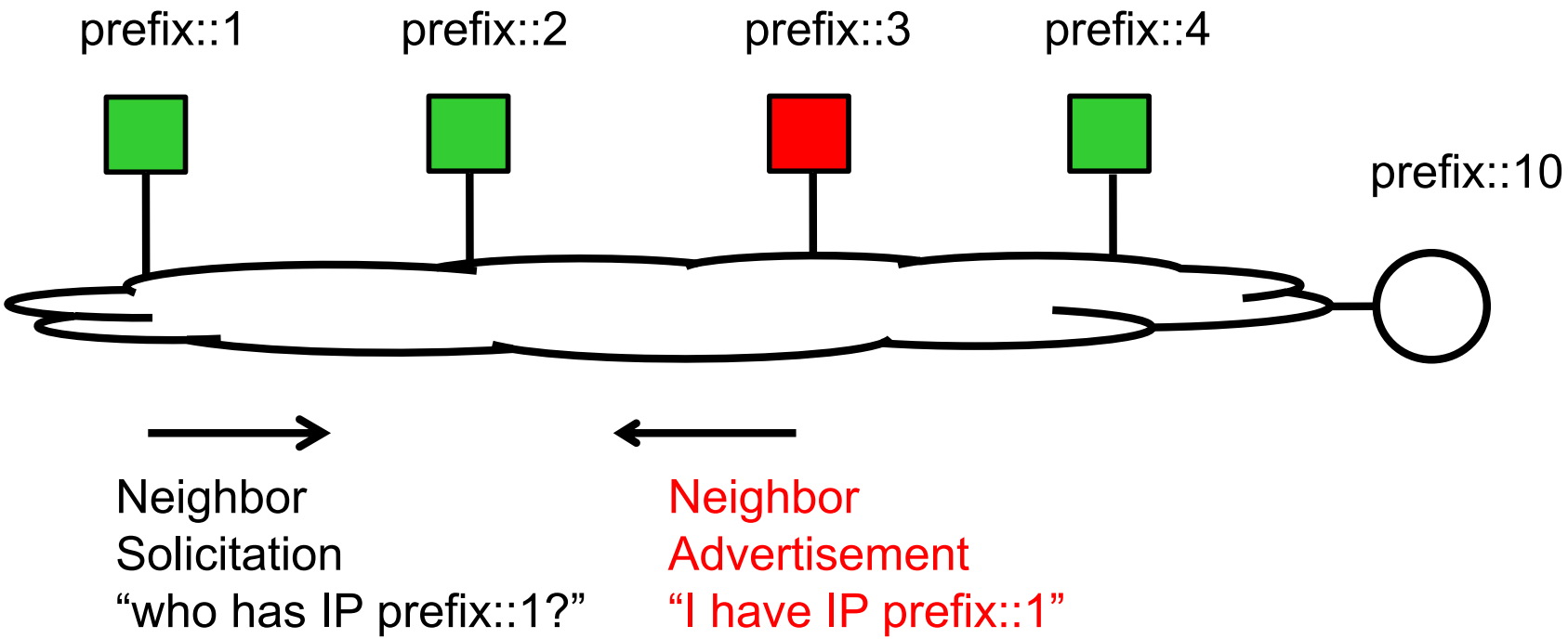


- no answer ➔ use address

Duplicate Address Detection Denial of Service

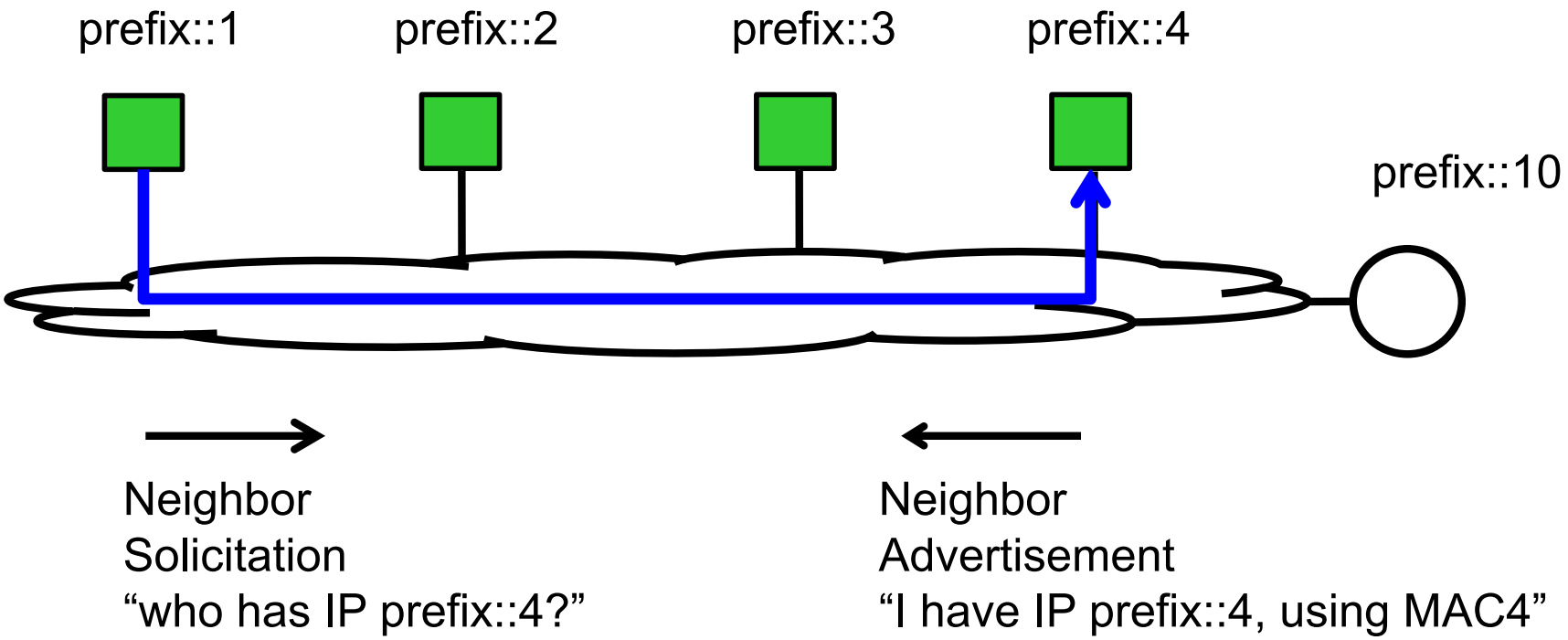
Malicious node in local network:

- claims to have the inquired IP address
- ➔DAD fails ➔ autoconfiguration fails



Address Resolution

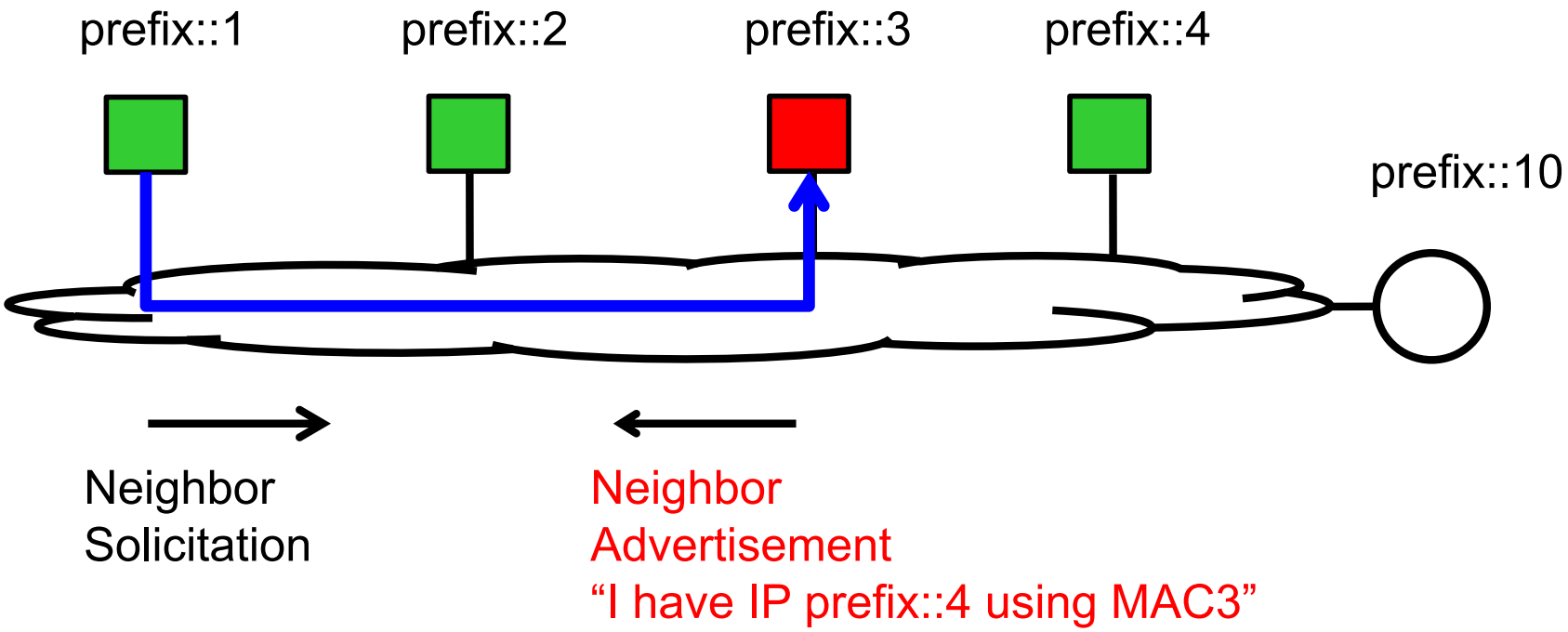
Purposes: get link-layer address of destination



Neighbor Solicitation/Advertisement Spoofing

Malicious node in local network:

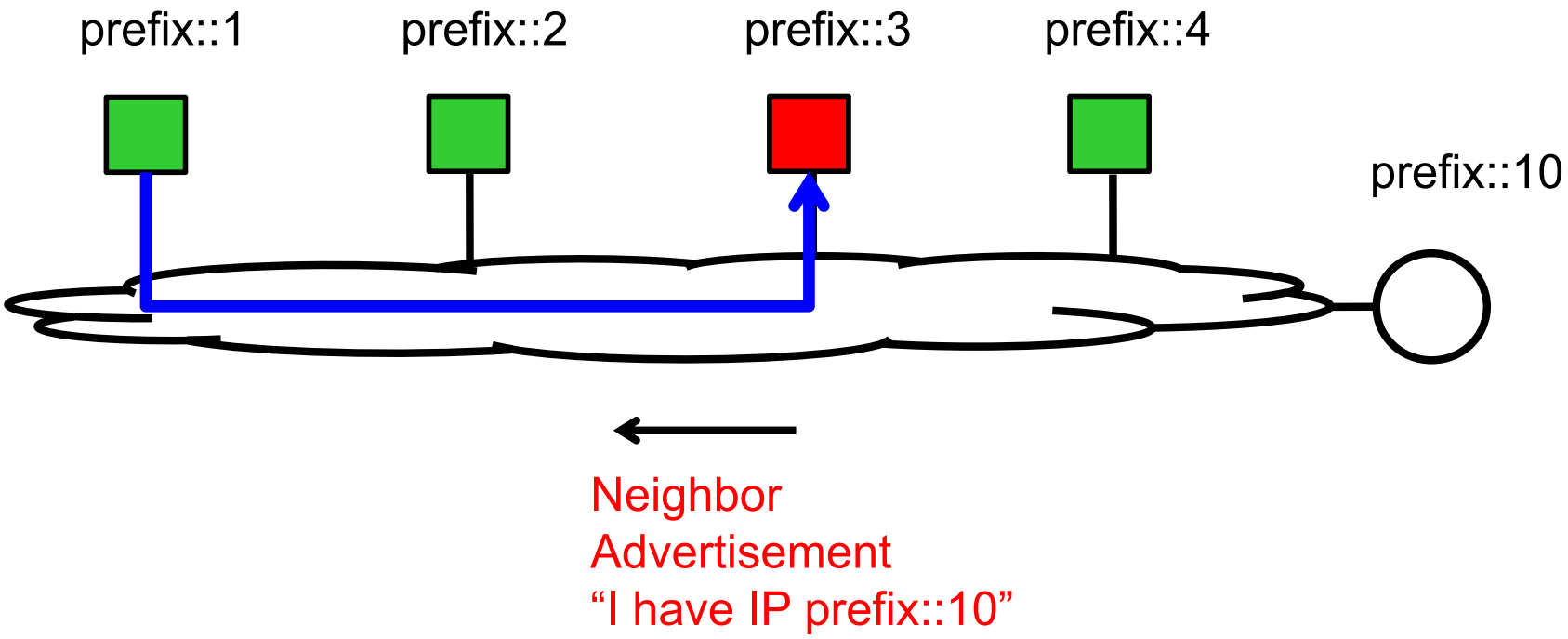
- can claim to have IP address of another node



Neighbor Solicitation/Advertisement Spoofing

Malicious node in local network:

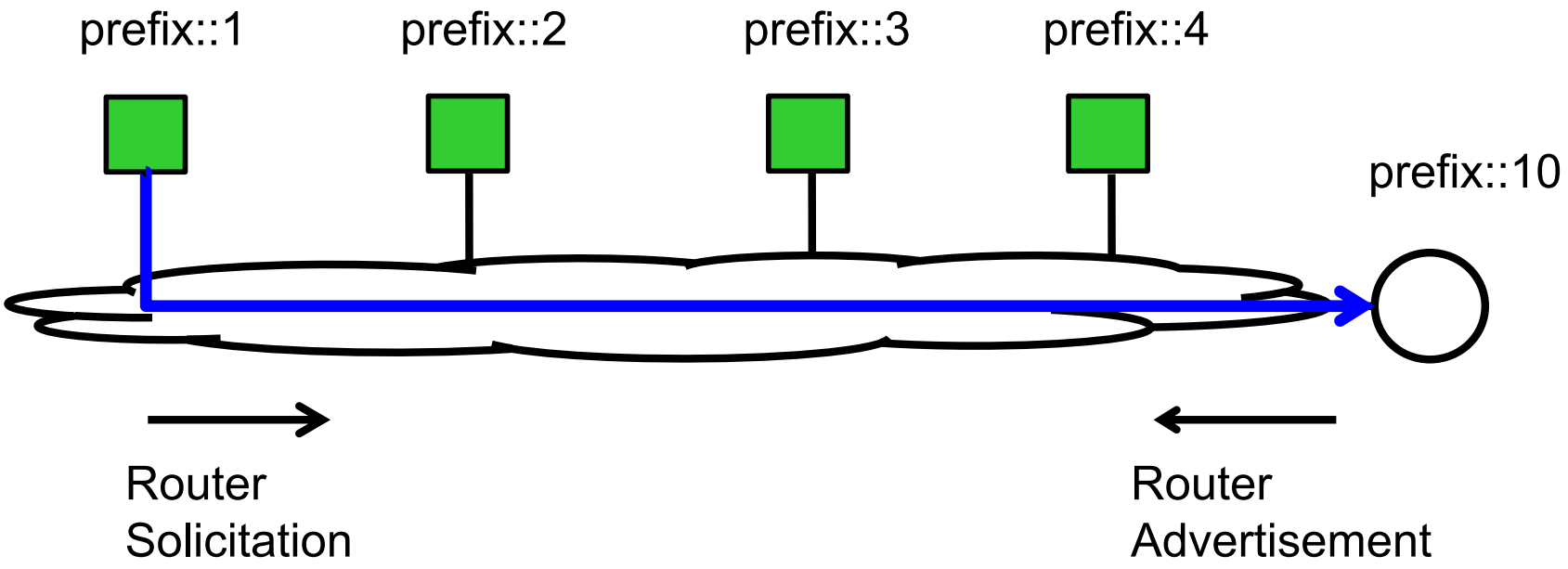
- can claim to have IP address of router



Router Solicitation

Purpose:

- Ask for Router Advertisement
- Learn local routers



Rouge Router

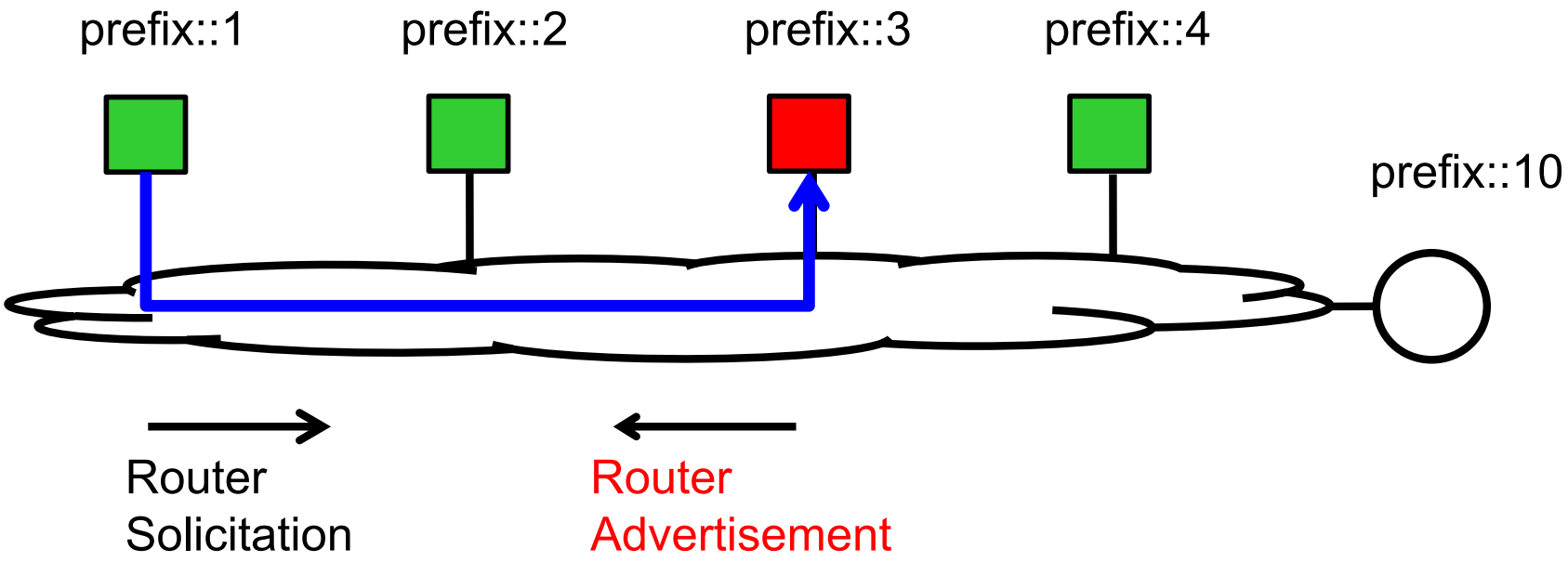
- Malicious node pretends to be router
  - Spoof Routing Advertisements
  - Injects wrong information

- Multiple methods to attack
  - Send wrong address prefix
  - Claim to be last hop router
  - Redirect traffic
  - Eavesdrop on traffic

Malicious Last Hop Router

Malicious node in local network:

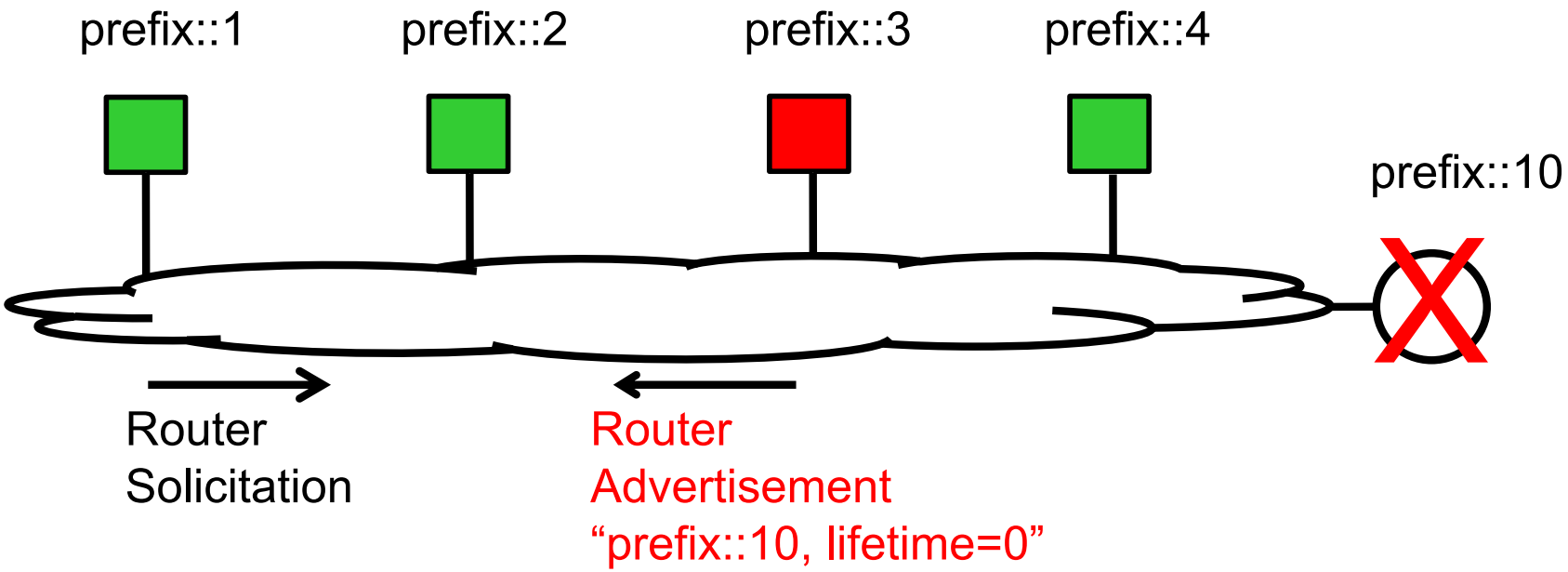
- can claim to be local router



Router Solicitation

Malicious node in local network:

- Can claim that existing router is no longer active



Securing NDP

- First approach in RFC 2461 (substituted by RFC4861)
  - Proposes to use IPsec
  - “NDP packet exchanges can be authenticated using the IP Authentication Header”

Problem: Key Distribution

- Manually
  - ➔ o.k., but does not scale
- Internet Key Exchange (IKE)
  - NDP needs IKE to secure messages
  - IKE needs IP addresses to establish security association
  - IPv6 nodes needs NDP to get IP address
  - ➔ chicken-egg problem

Secure Neighbor Discovery (SEND) [RFC3971]

- Purpose: prevent attacks based on NDP
  - Make NDP usable in environments were nodes may not be trusted

- Secure NDP:
  - Ensure message integrity
  - Prevent address theft
  - Prevent replay attacks
  - Support router authorization

Building Blocks

- Asymmetric cryptography
  - Each node has key pair (private/public)
- Digital Signature method
  - Using key pair
- Certificates
  - Assuring correct mapping of public key to identitiy

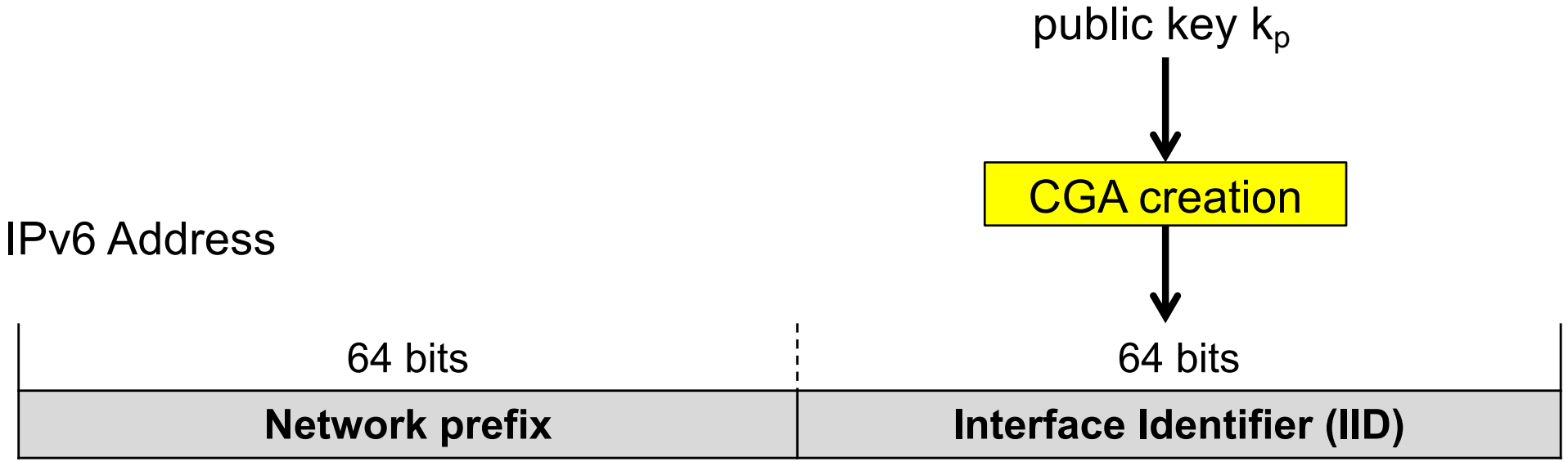
Building Blocks

- Cryptographically Generated Addresses (CGA)
  - Bind an address to a public key
- RSA Signature
  - Protect Neighbor and Router discovery messages
  - New option in NDP messages
- Timestamp and Nonce
  - Prevent replay attacks
- Certification Paths
  - Certification Path Solicitation/Advertisements
  - Authorize a node to act as router
  - Requires a trust anchor

Cryptographically Generated Addresses (CGA)

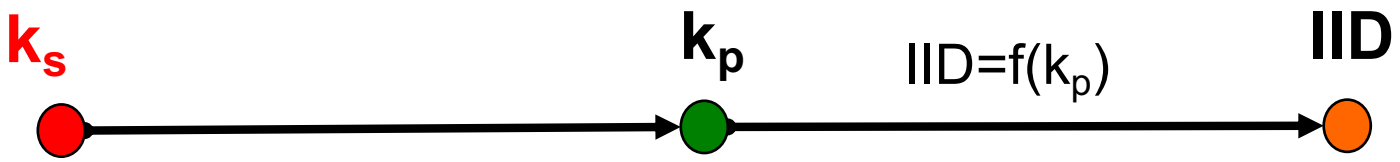
Cryptographically Generated Addresses (CGA)

- Idea: Bind public key to IPv6 address [RFC 3972]
  - Derive IPv6 interface identifier (IID) from public key
- Host generates keypair k_s, k_p
  - Uses public key to generate IID



Purpose of CGA

- Bind public key to Address



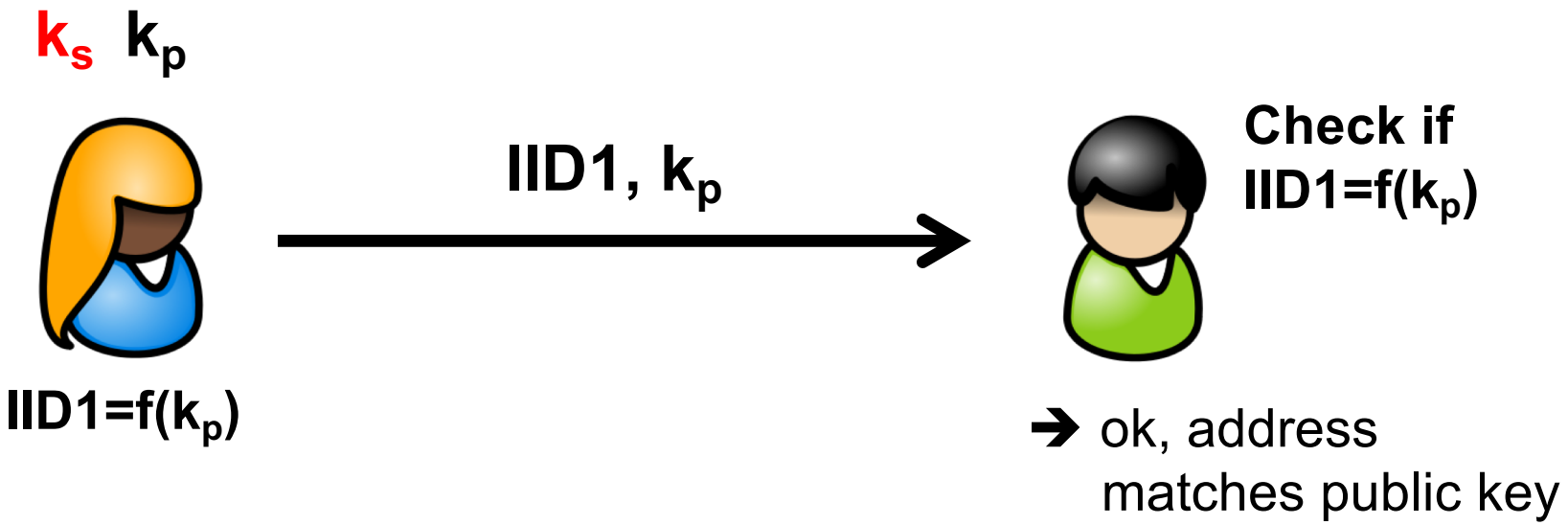
Only secret key k_s matches to k_p
Only messages signed with k_s can be verified with k_p

key k_p is bound to IID

* Multiple keys can generate this IID, but we make it hard to find a another k'$_p$ that can generate the same IID

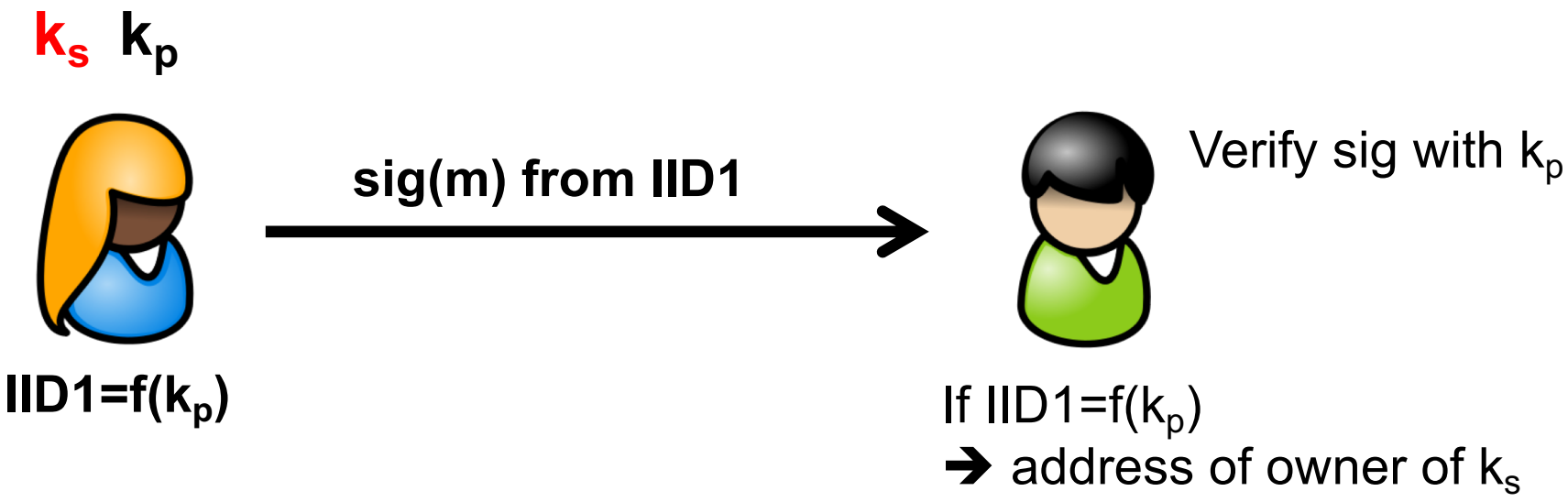
Cryptographically Generated Addresses (CGA)

- Idea: Bind public key to IPv6 address [RFC 3972]
  - Receiver can check if address matches the public key
  - Uses public key of sender to generate IID
  - If own generated IID matches IID in address ➔ public key matches the address

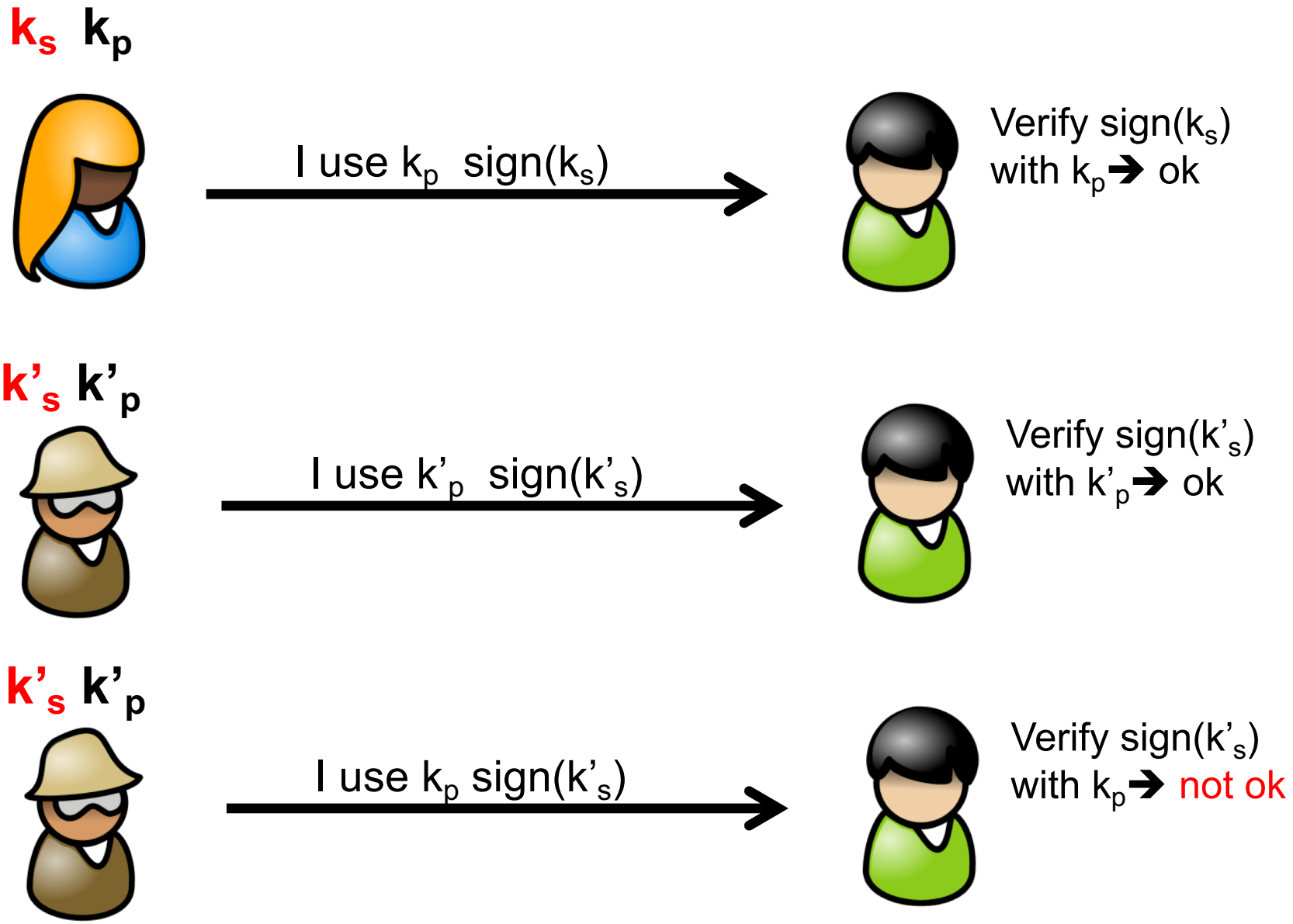


Cryptographically Generated Addresses (CGA)

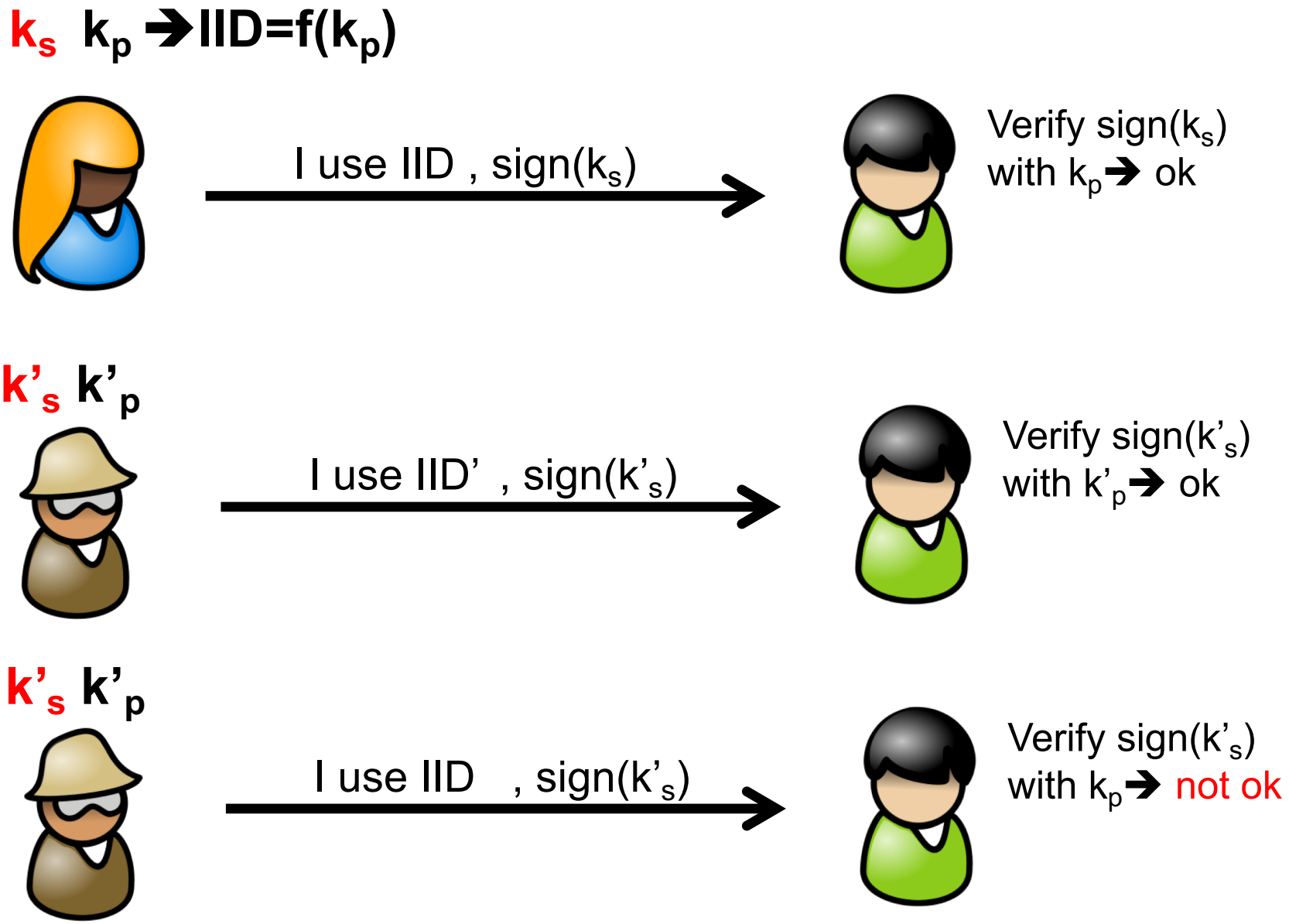
- Sender uses private key to sign NDP messages, e.g.
  - Neighbor Advertisement
  - Router Advertisement
- Receiver verifies signature with public key
  - Checks if signature can be validated with k_p ➔public key belongs to host that signed
  - If IID= $f(k_p)$➔ address belongs to host that signed



Attacker cannot claim to use Same Public Key



Attacker cannot Claim to use Same Address



Cryptographically Generated Addresses (CGA)

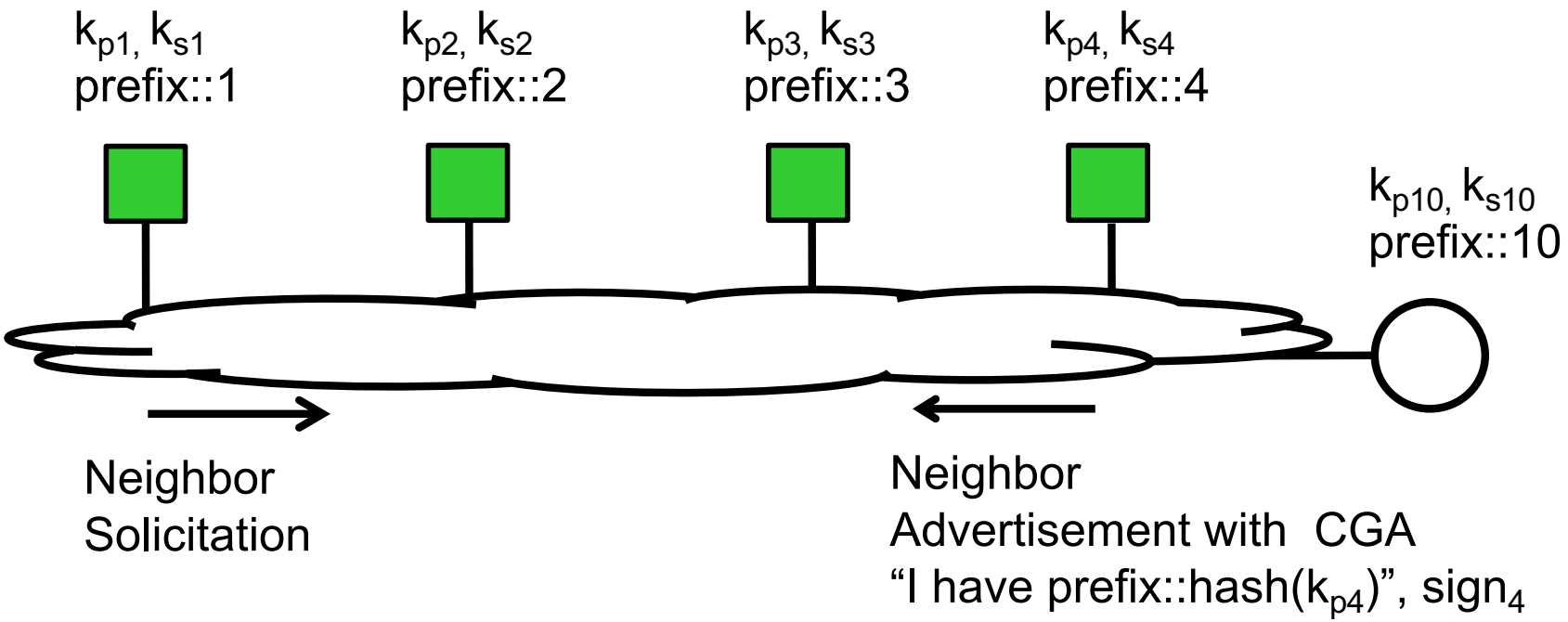
- Malicious Node does not know k_s
  - Can generate IID=f(k_p), but cannot sign with k_s
  - Can generate own address IID'=f(k'_p) and sign messages from IID' with own k'_s
  - But cannot generate a valid signature for messages from IID
  - ➔ signature from attacker cannot be verified with k_p

➔ Attacker can generate CGA but cannot send signed messages with CGA of other node

k'_s k'_p

IID'=f(k'_p)
IID=f(k_p)

Secure Neighbor Discovery



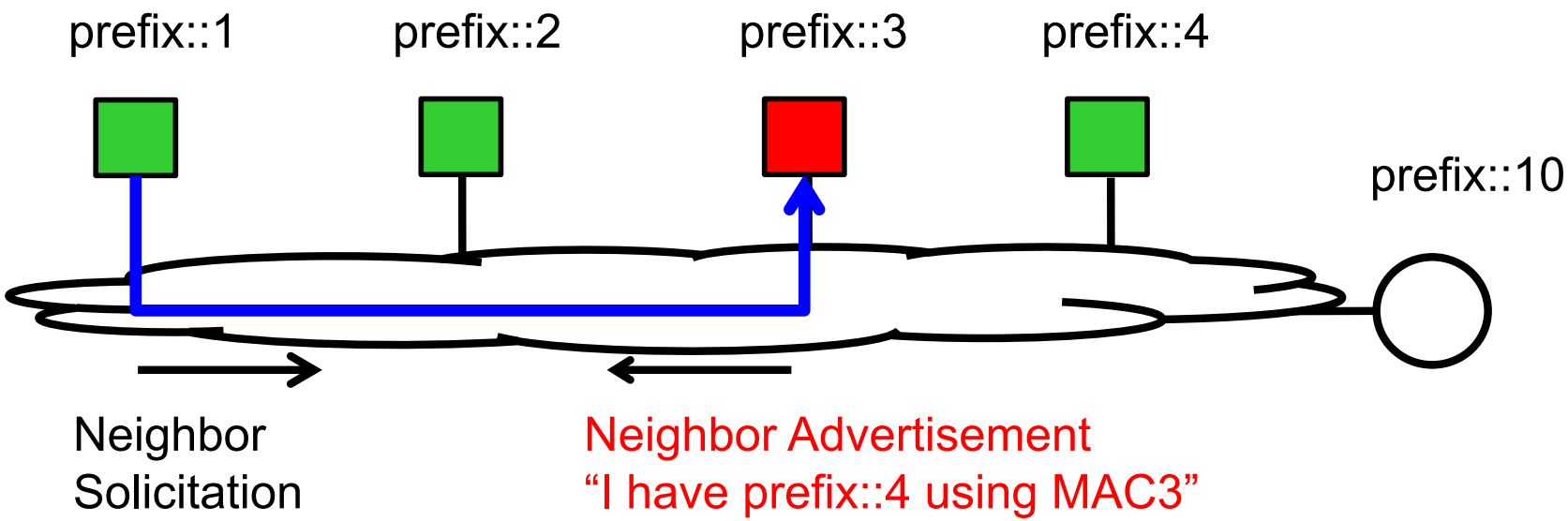
- Each node has private/public keypair
- Generates CGA based on public key (CGA option)
- Signs message with private key (RSA signature option)

Neighbor Advertisement Spoofing (without CGA)

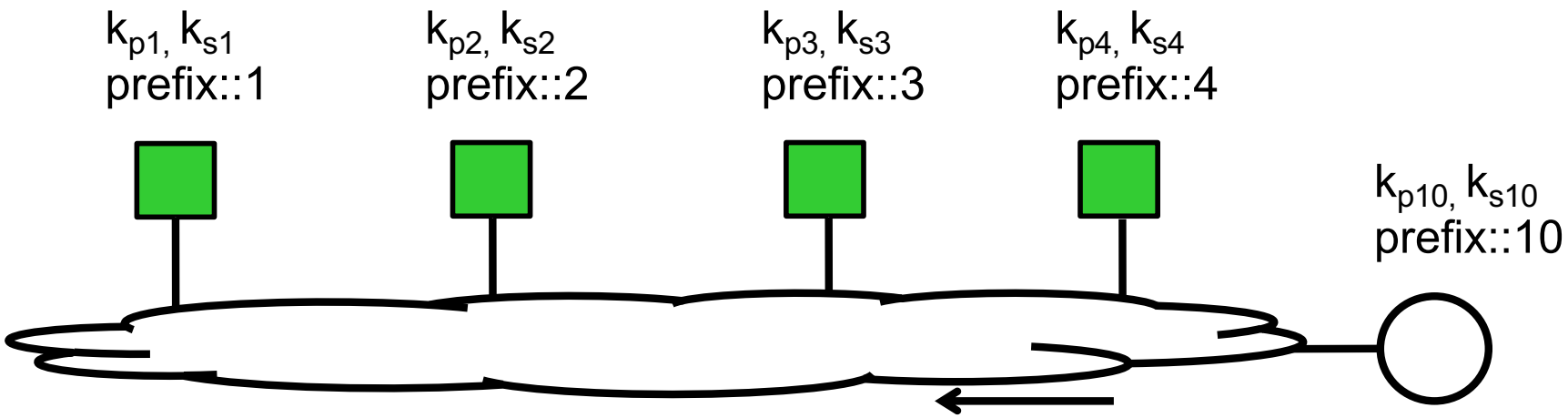
Malicious node in local network:

- can claim to have IP address of another node
- Like ARP spoofing in IPv4

Without CGA:



CGA: Secure Neighbor Discovery



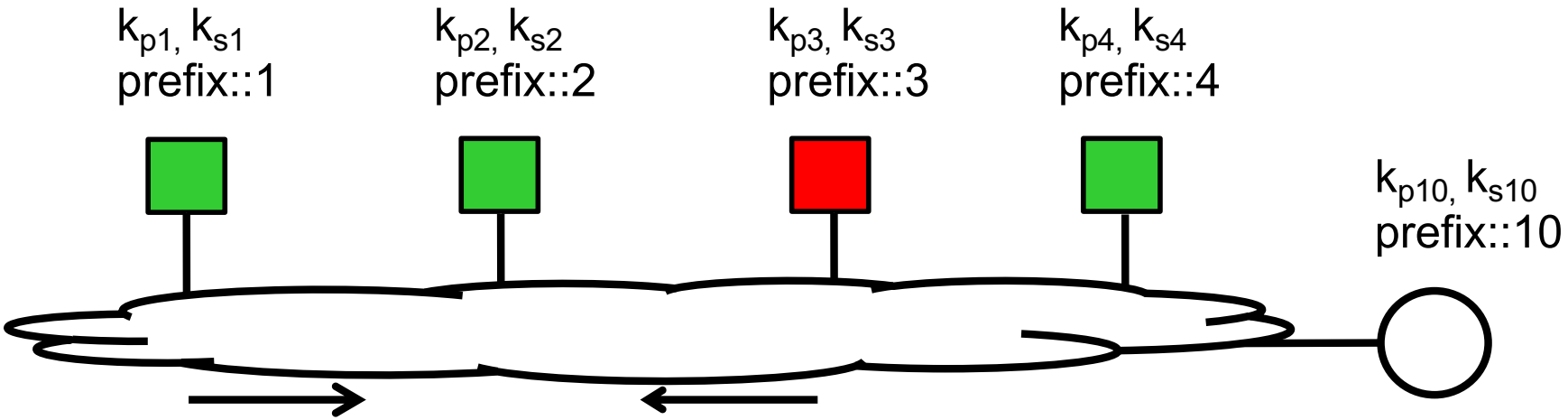
Verify signature with k_{p4}
Calculate hash(k_{p4})
Compare hash(k_{p4}) with IP address

Neighbor
Advertisement with CGA
"I have prefix::hash(k_{p4})", $sign_4$

➔ receiver can check if signature matches address

CGA: Prevent Neighbor Advertisement Spoofing

With CGA: Sender of (signed) NDP message is owner of claimed IID



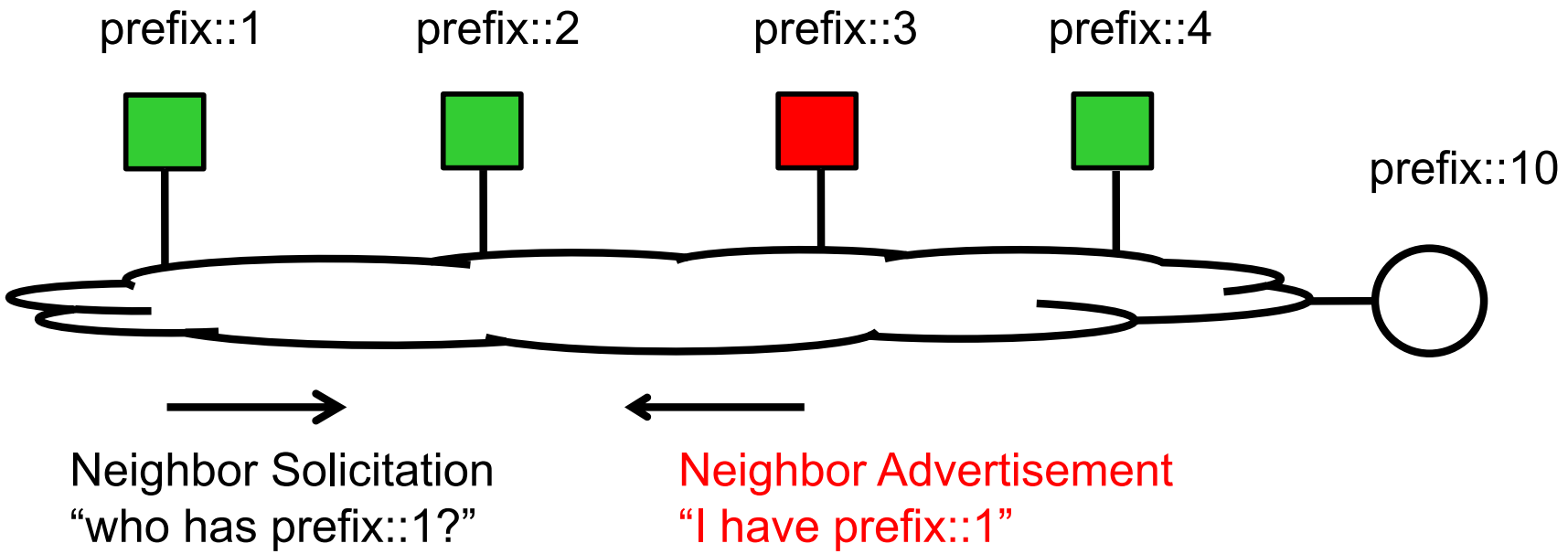
- Signature and address do not match
  - Malicious node cannot sign with private key of other node
  - Malicious node can generate CGA, but cannot claim to have IP of a different node

Without CGA: DAD Denial of Service

Malicious node in local network:

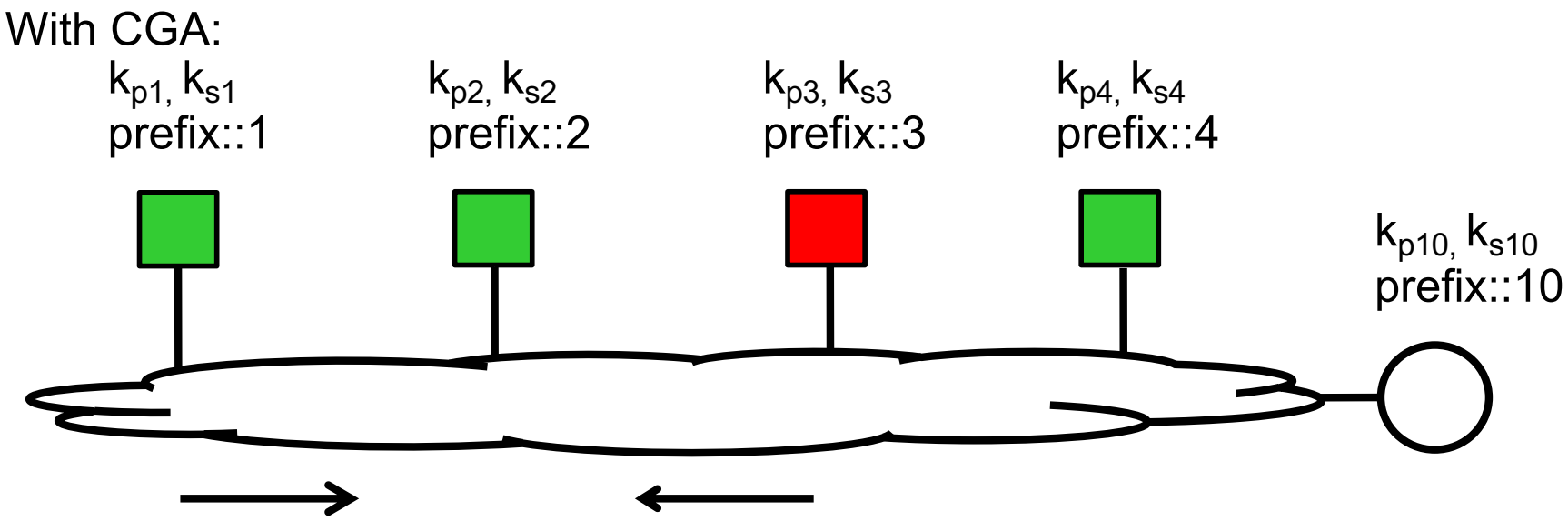
- Duplicate Address Detection (DAD)
- Malicious node claims to have the inquired IP address
- ➔DAD fails ➔ autoconfiguration fails

Without CGA:

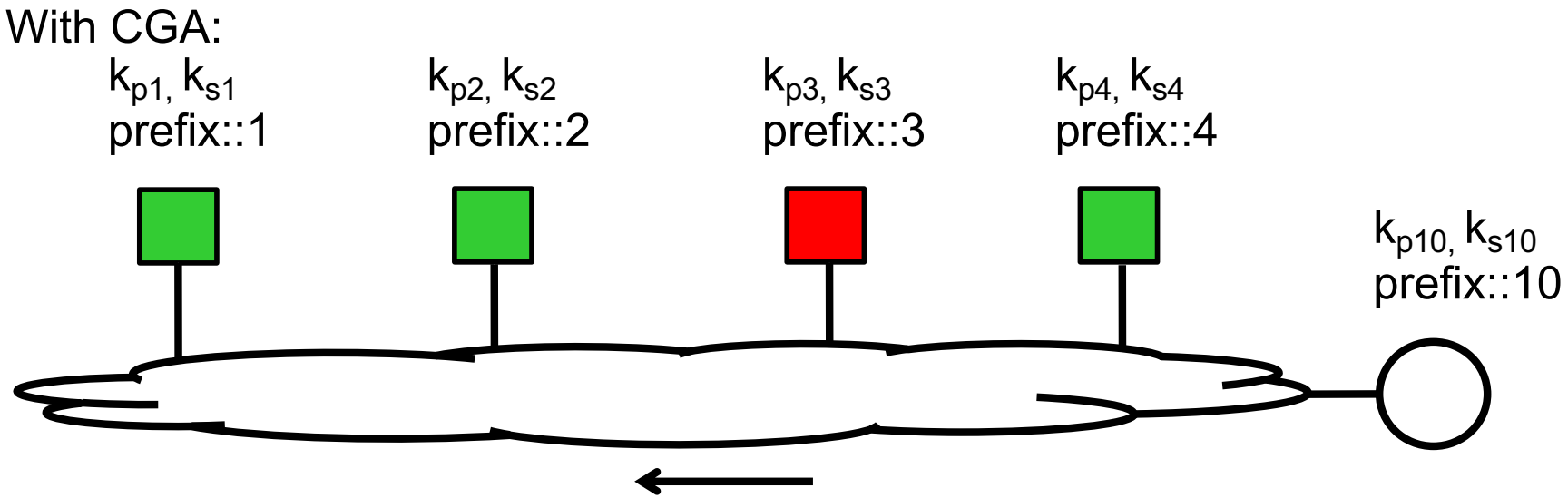


CGA: Prevent DAD DoS

Sender of (signed) NDP message is owner of claimed IID



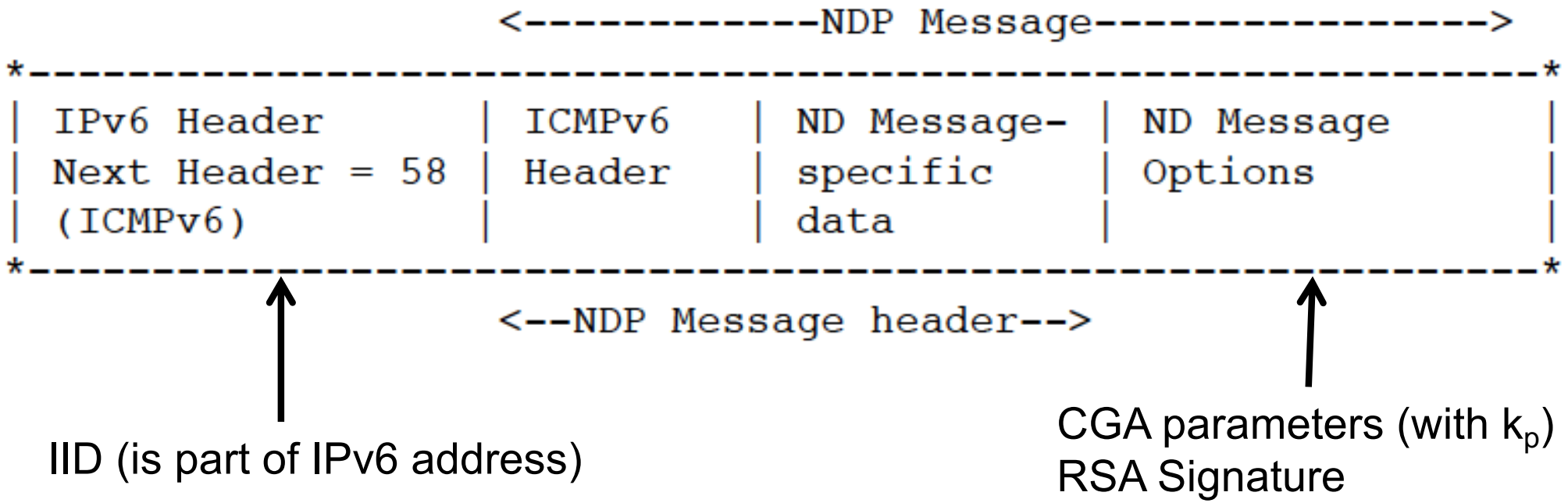
Prevent wrong Router Announcements?



not prevented with CGA as long as address ok

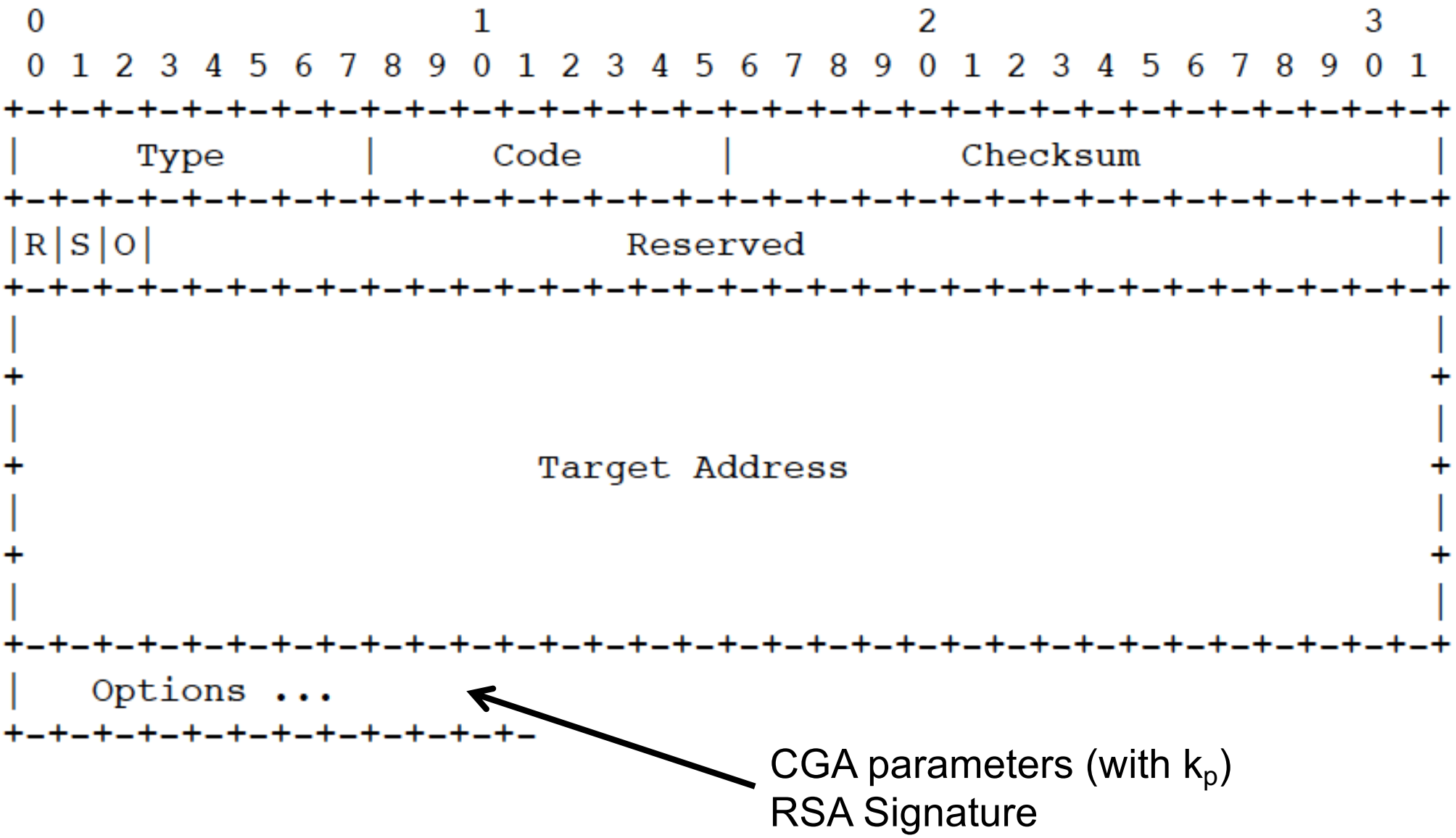
➔ Router Authorization

NDP Message Format [RFC3971]



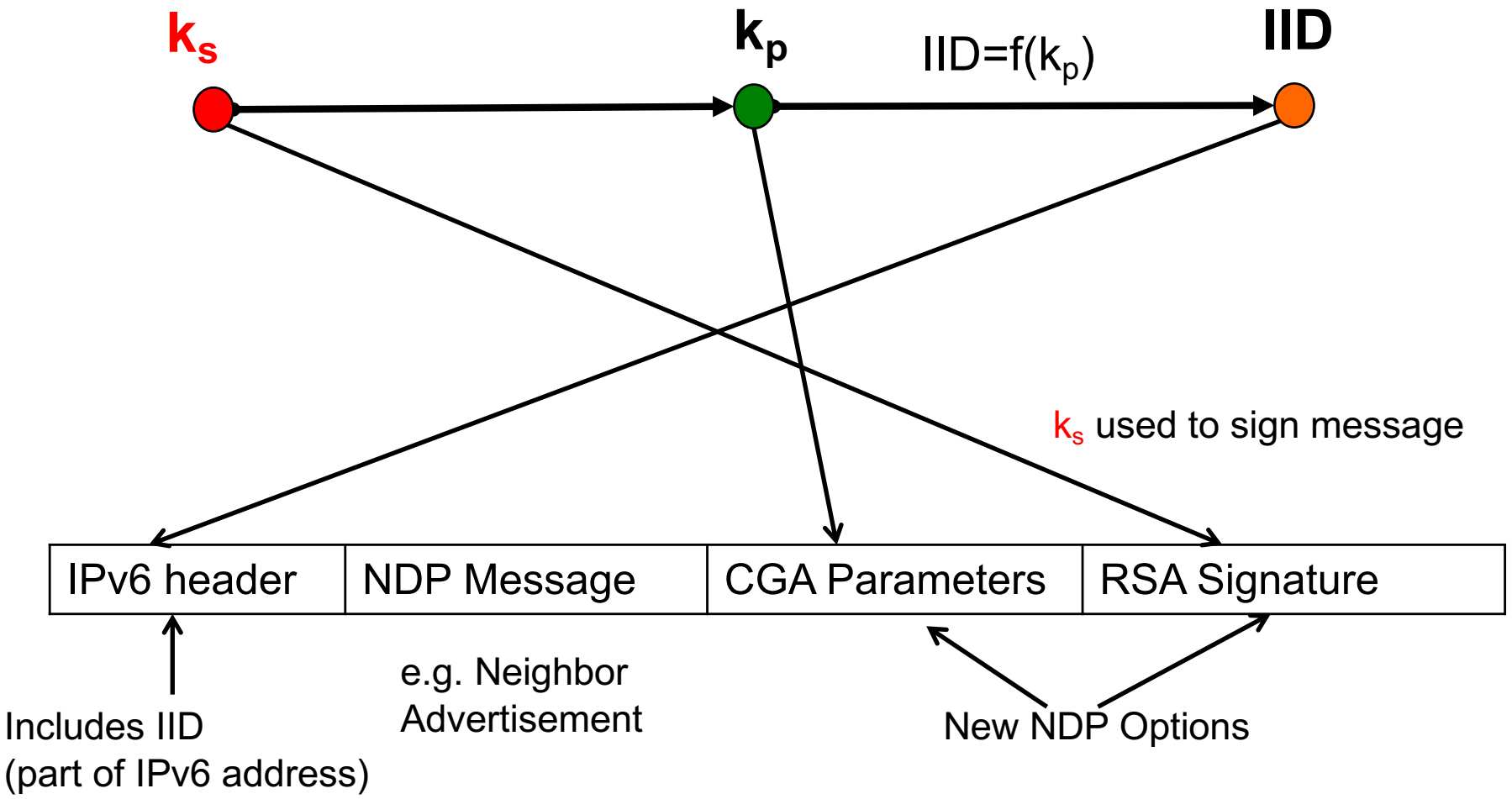
- New NDP Options
  - CGA Parameters
  - RSA Signature

Neighbor Advertisement Message [RFC4861]



NDP Message

- NDP Message



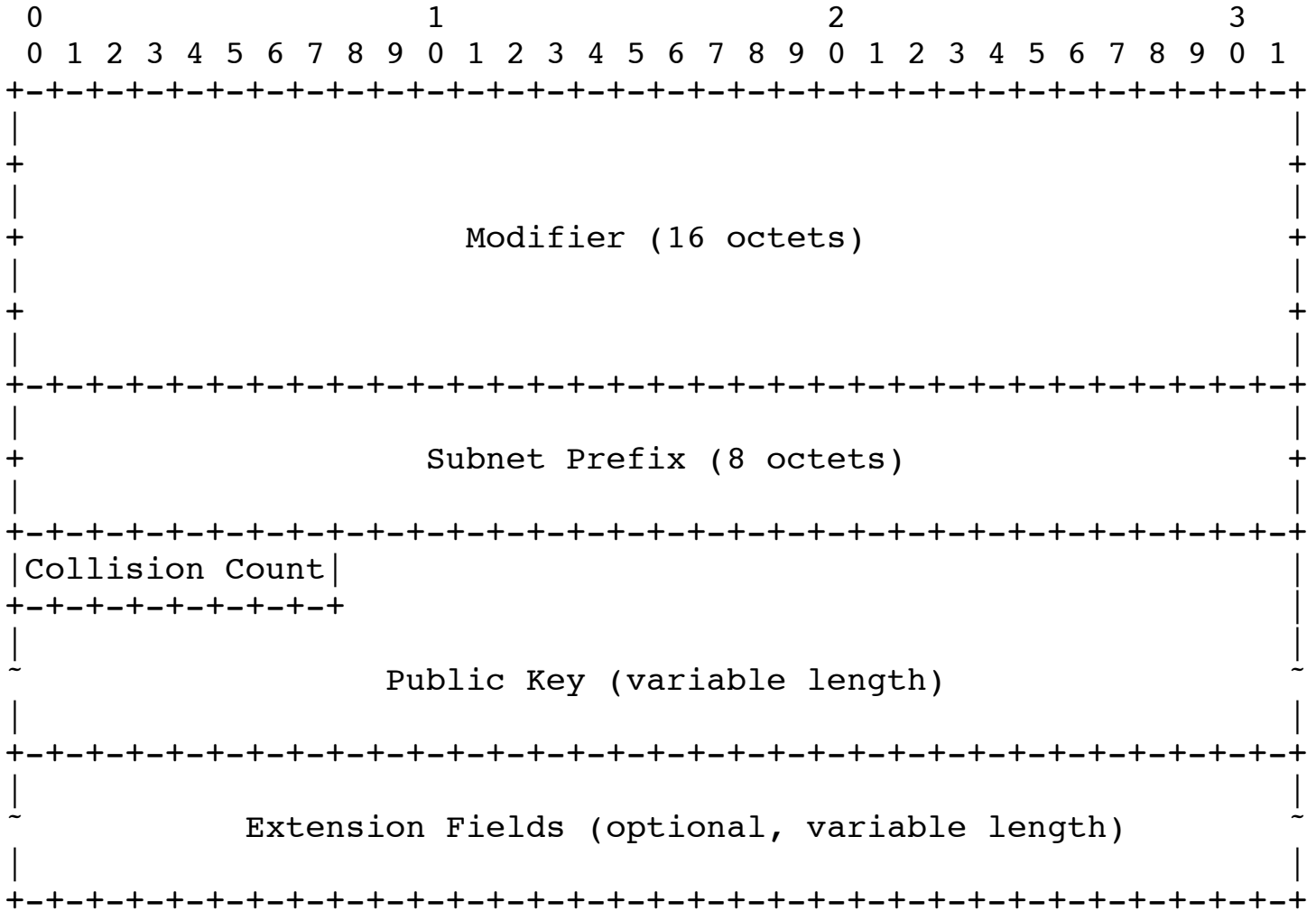
* Multiple keys can generate this IID, but we make it hard to find a another k'$_p$ that can generate the same IID

CGA Parameters Data Structure

CGA is calculated from those parameters

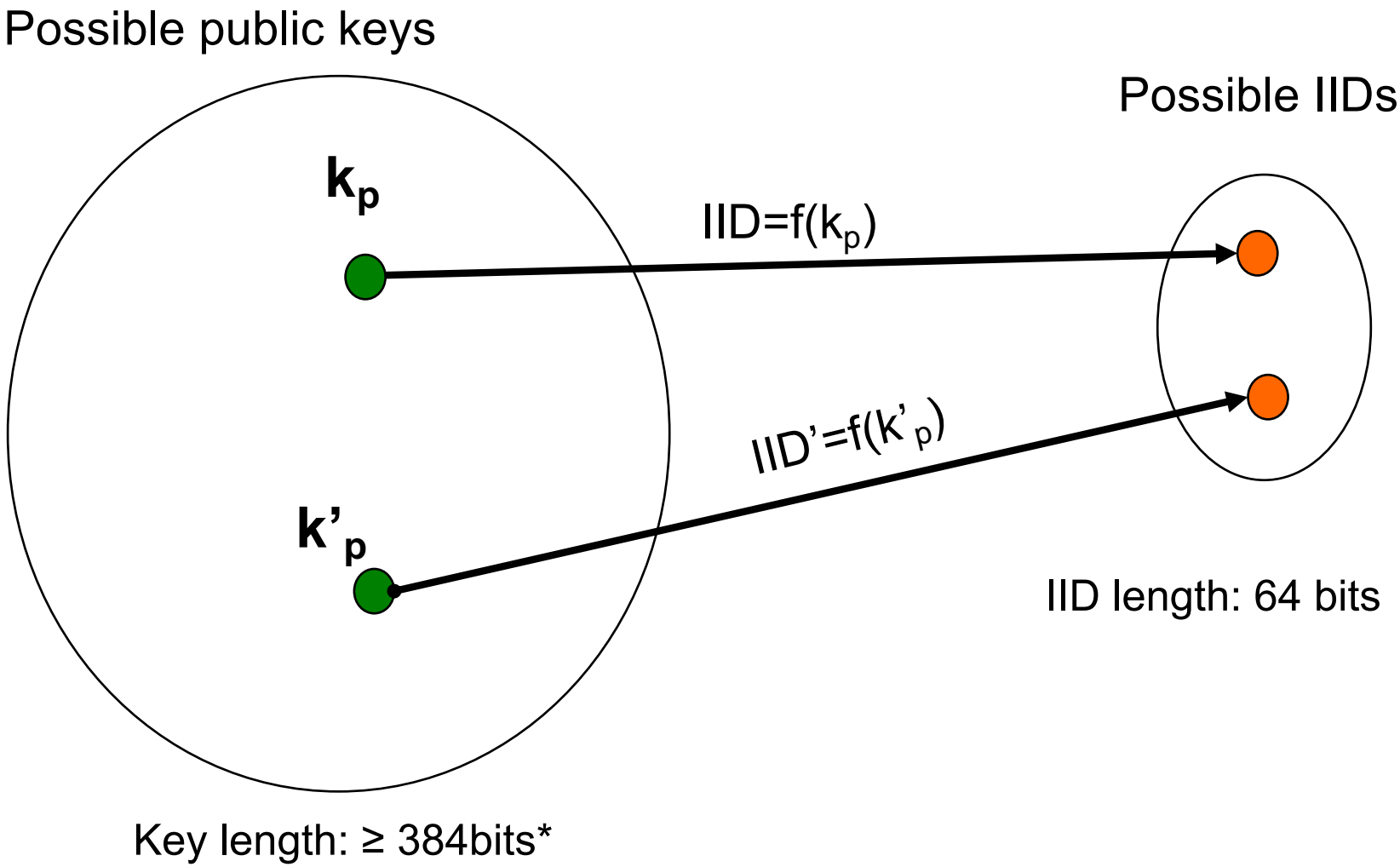
arbitrary integer to add randomness

Increased if DAD collision



Source: RFC3972

How should CGA Generation Function look like?



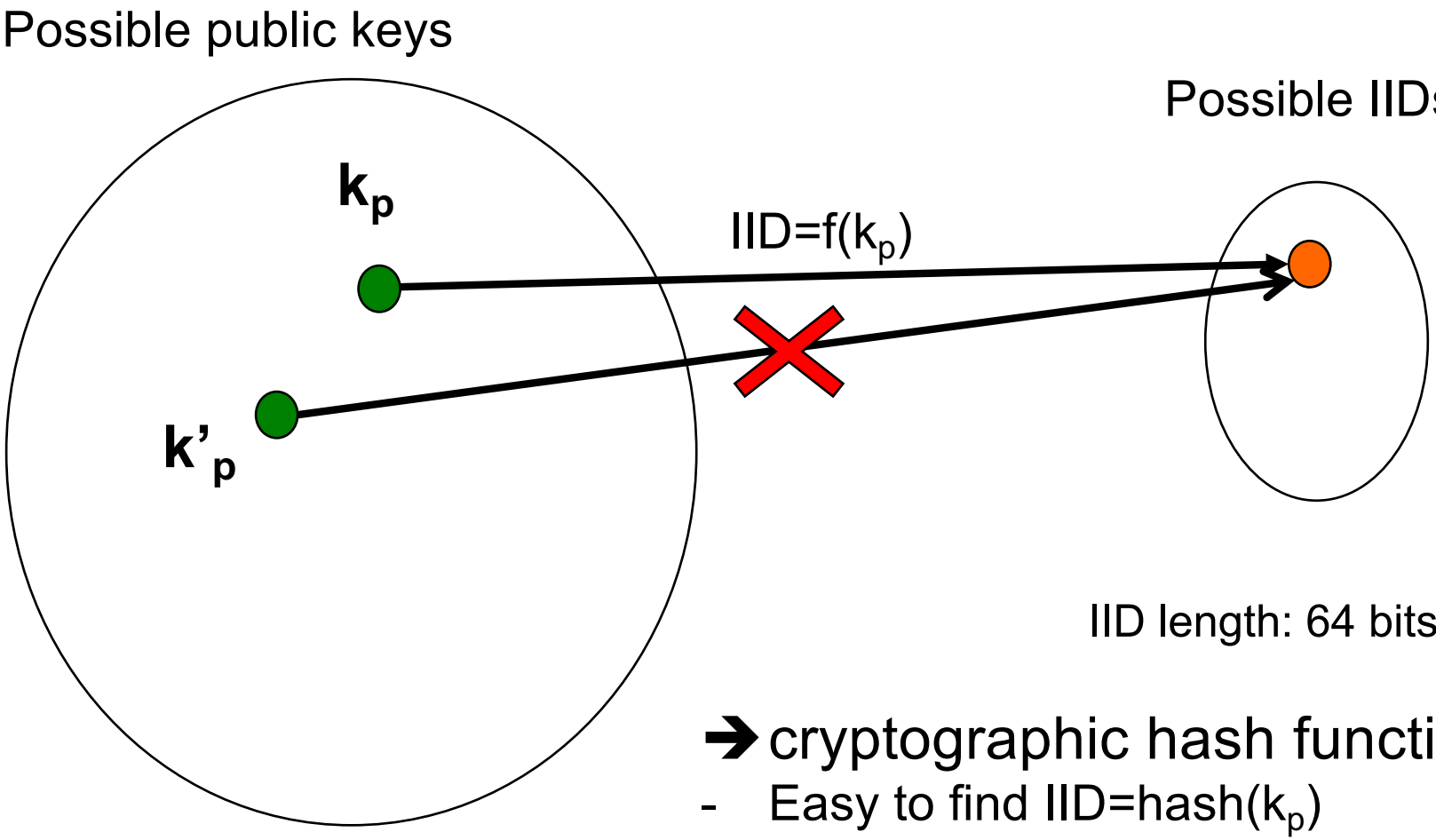
* [RFC3972], but too short for strong authentication/secrecy

How should CGA Generation Function look like?

- What if attacker could find a public key k'_p that generates the same address? $IID'=f(k'_p)=IID=f(k_p)$
  - Could use his private key to sign messages for the victims address
  - $IID=f(k_p)$ ➔ should be hard to find a k'_p such that $IID=f(k'_p)$

Should not find a k'_p such that $IID=f(k'_p)=f(k_p)$

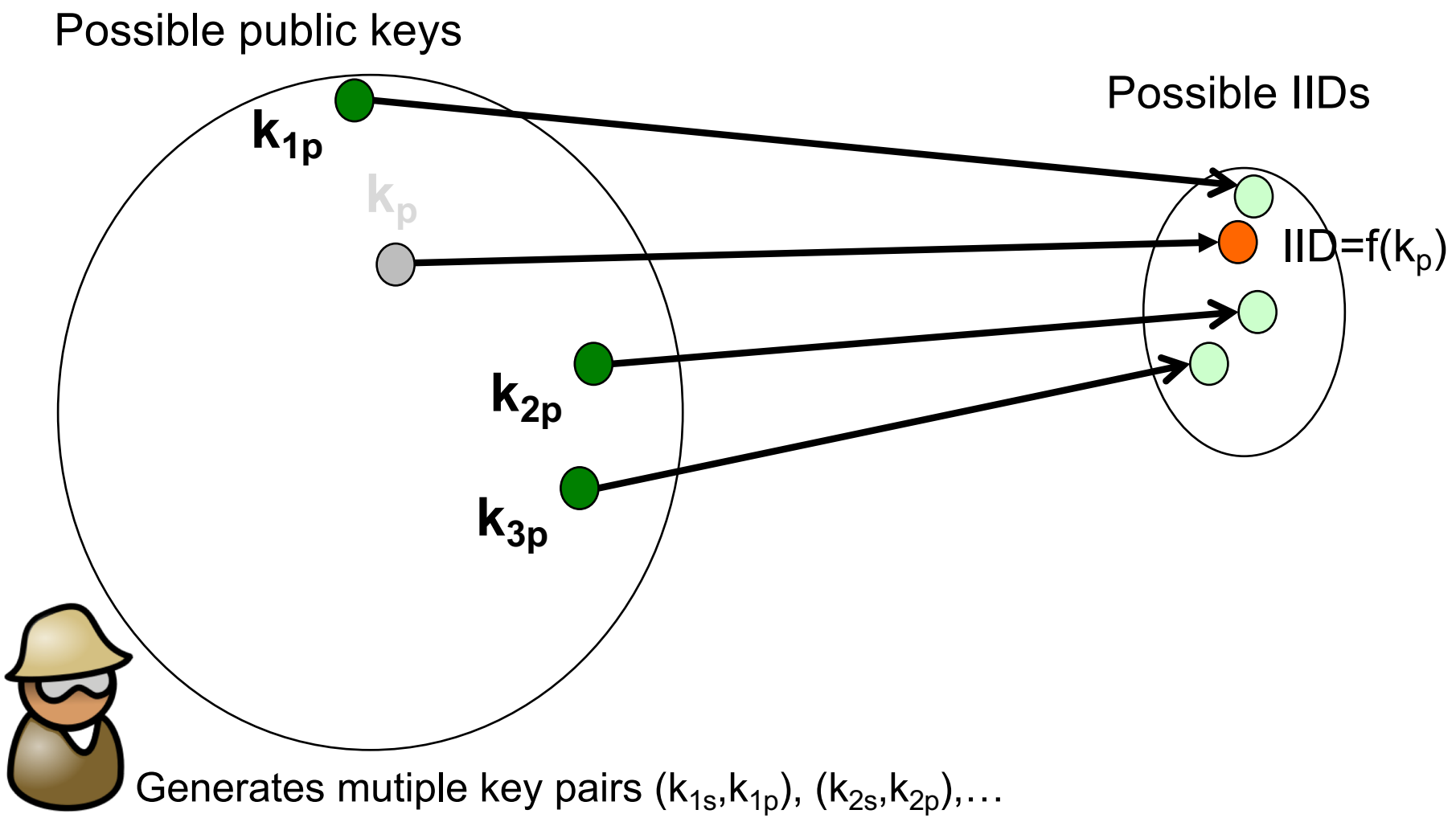
Prevent Attacker to find k'_p with IID=f(k'_p)=f(k_p)



➔ cryptographic hash function
- Easy to find IID=hash(k_p)
- Hard to find another k'_p with IID=hash(k'_p) (a ***hash collision***)

* [RFC3972], but too short for strong authentication/secrecy

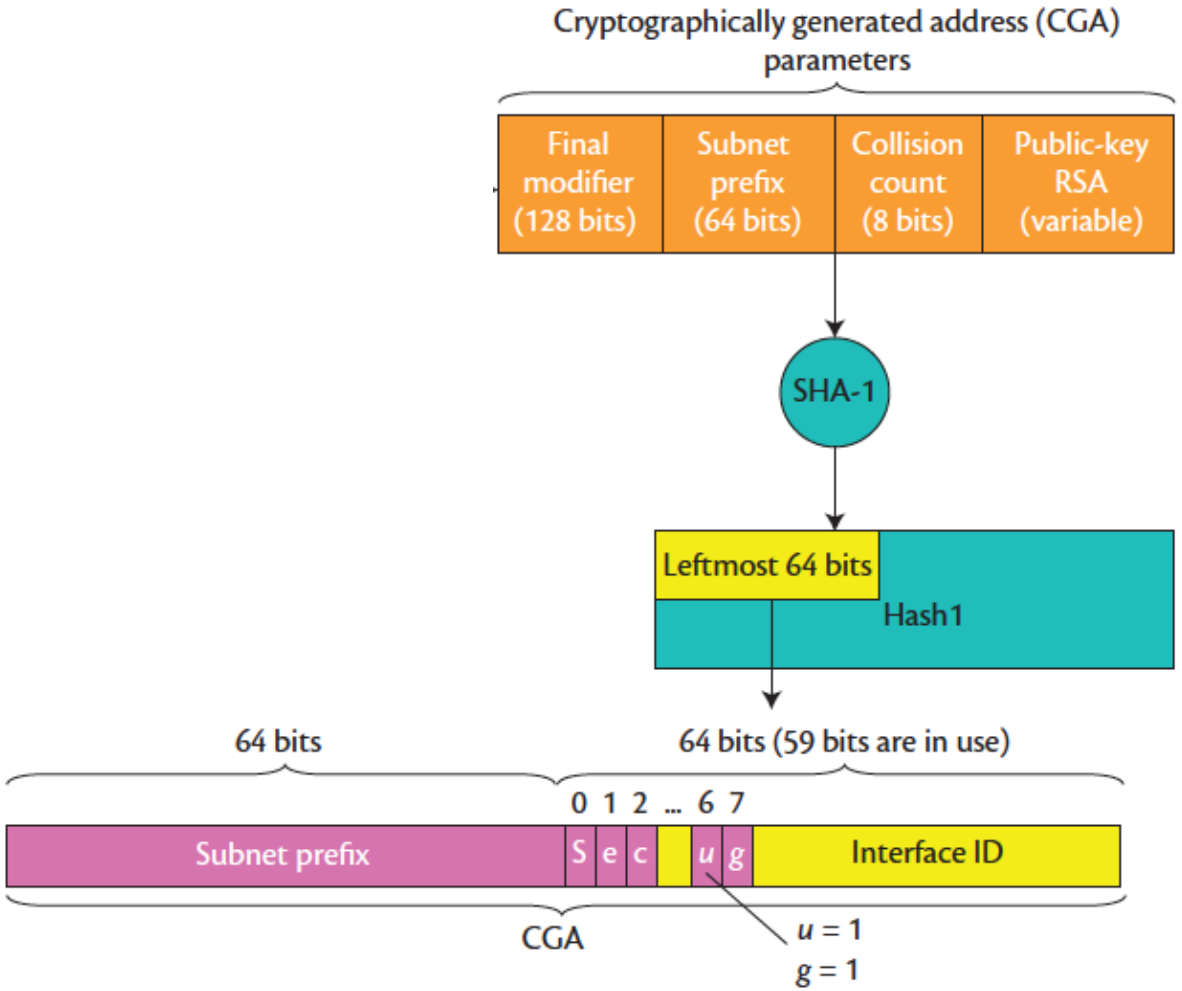
Brute Force



Generates mutiple key pairs (k_{1s},k_{1p}), (k_{2s},k_{2p}),…
Try different keys to find a hash collision

Always possible, but takes time ➔ computationally secure

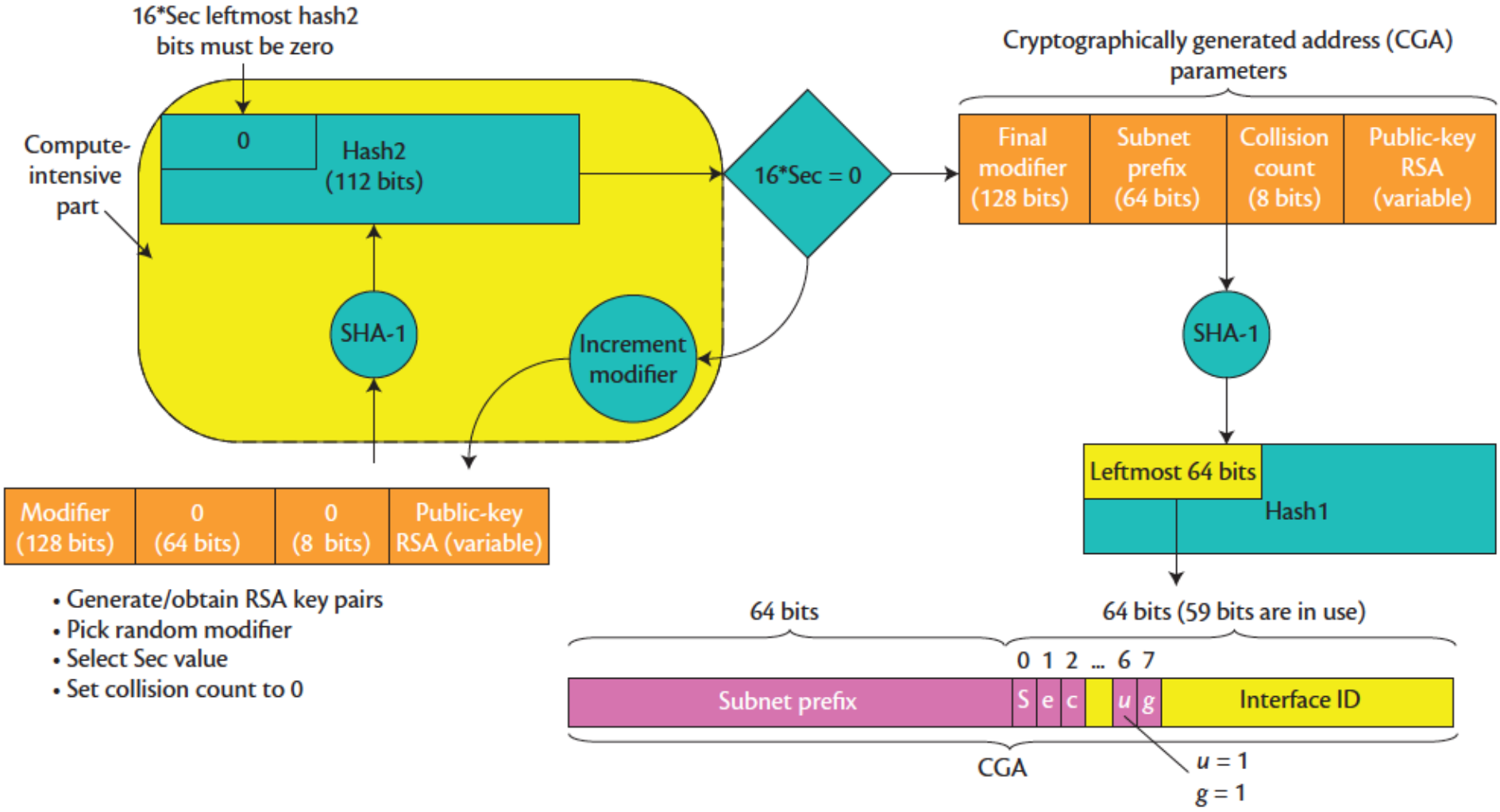
CGA Generation for Security Level Sec=0



Source: AlSa'deh, Meinel, "Secure Neighbor Discovery: Review, Challenges, Perspectives, and Recommendations," *IEEE Security Privacy*, vol. 10, no. 4, pp. 26–34, 2012.

CGA Generation for Sec>0

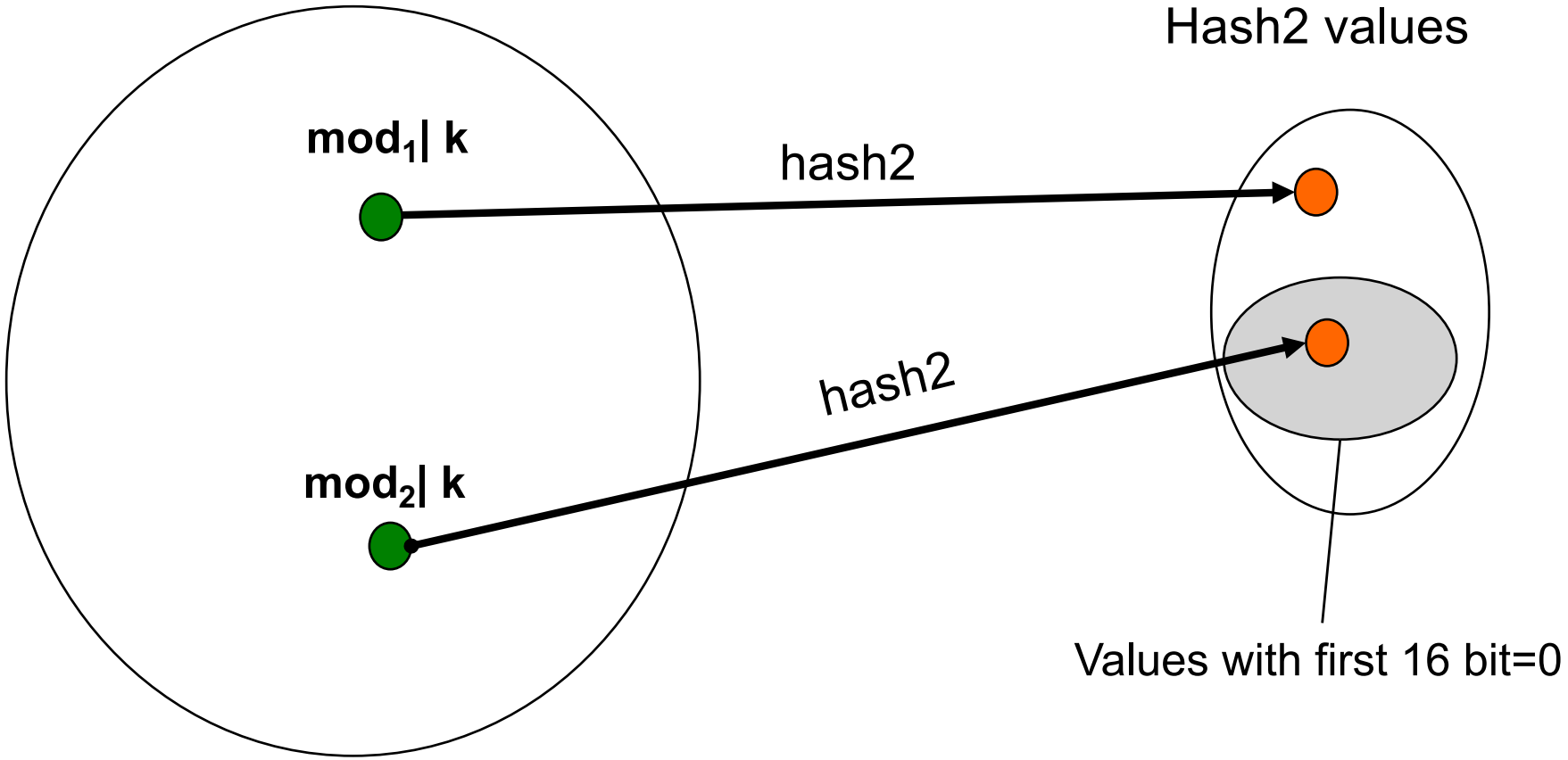
Hash Extension:



Source: AlSa'deh, Meinel, "Secure Neighbor Discovery: Review, Challenges, Perspectives, and Recommendations," *IEEE Security Privacy*, vol. 10, no. 4, pp. 26–34, 2012.

Hash2 Loop (if Sec>0)

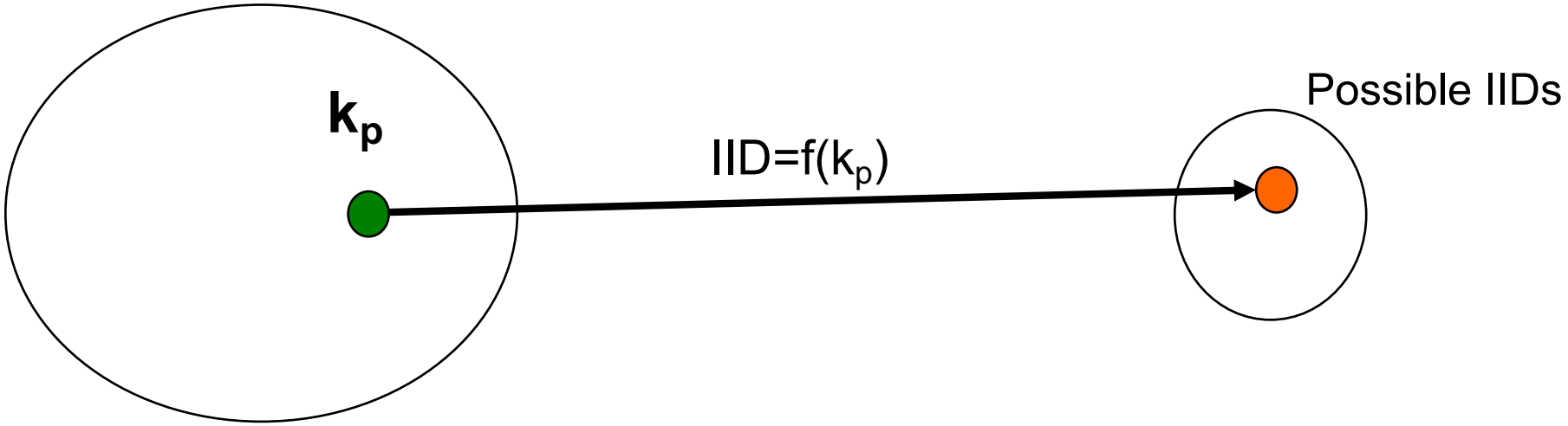
Input: modifier || public key



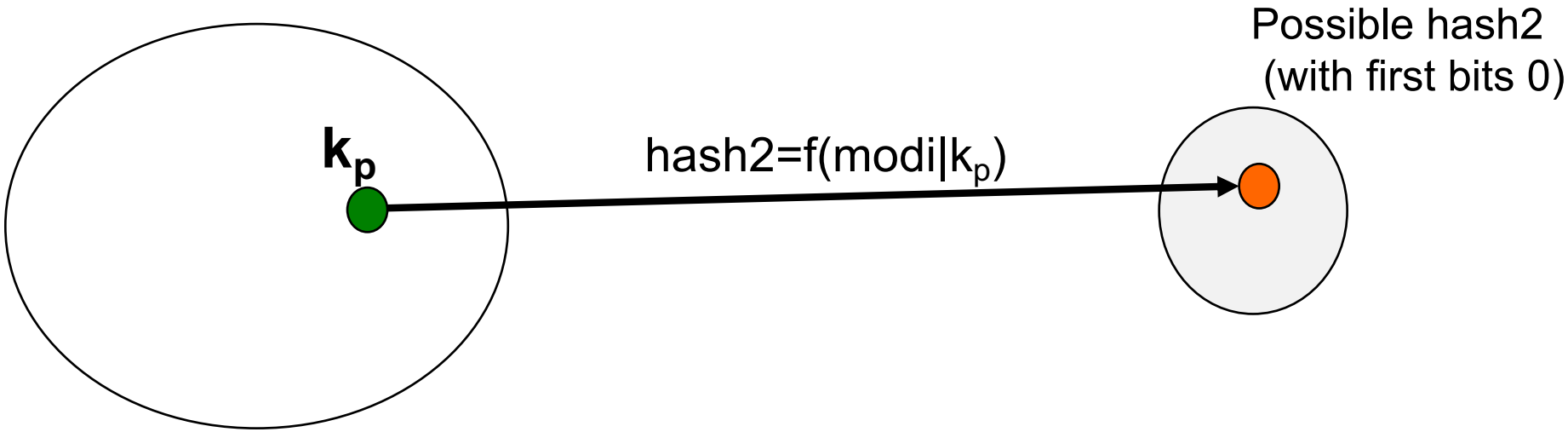
Search (modify modifier) until hash output in target zone

Sec >0

Attacker needs to find k'_p that a) Condition1: generates correct IID



AND b) Condition 2: generates a valid hash2 with the given modifier



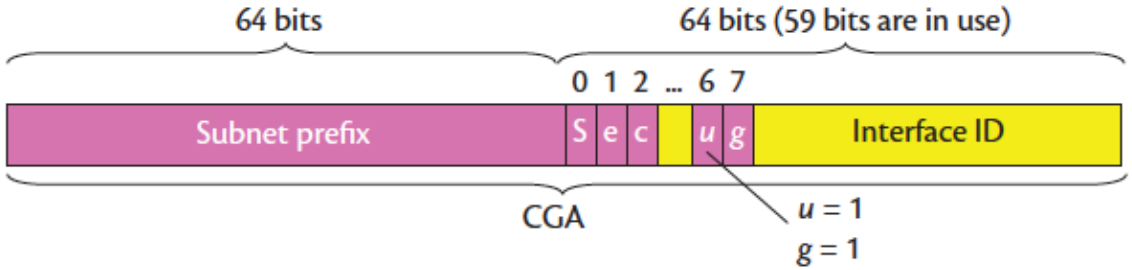
CGA Generation

- Receiver needs modifier
  - to validate CGA
- ➔ modifier sent in CGA parameters

- ➔ Attacker knows modifier

- But now 2 conditions must hold
  - Condition 1: hash1(public key, CGA parameters) ➔ IID
  - Condition 2: hash2 (modifier || public key) ➔ first bits = 0
- Harder to find a hash collision

Receiver Operation



- `Mask1 (64 bits) = 0x1cffffffffffffff`
- `Mask2 (112 bits) =`

```
0x00000000000000000000000000000000 if Sec=0
0xffff00000000000000000000000000000 if Sec=1
0xffffffff00000000000000000000000000 if Sec=2
0xffffffffffff000000000000000000000 if Sec=3
0xffffffffffffffff0000000000000000 if Sec=4
0xffffffffffffffffffff00000000 if Sec=5
0xffffffffffffffffffffffff0000 if Sec=6
0xffffffffffffffffffffffffffff if Sec=7
```

- Receiver checks if

& ➔ logical AND

`Hash1 & Mask1 == IID & Mask1`

`Hash2 & Mask2 == 0x00000000000000000000000000000000`

CGA Generation

CGA Generation times based on tests

Device specifications	Sec = 0	Sec = 1	Sec = 2	Sec = 3
Modern PC (AMD64)[10]	n/a	0.2 s	3.2 hours	n/a (theoretically, 24 years)
Machine with 2.67-GHz CPU[11]	93.41 ms	401.99 ms	1.65 hours	n/a (theoretically, 12 years)
Duo2 (2.53-GHz) workstation[12]	100 μs	60 ms	2,000 s	n/a (theoretically, more than 30,000 hours)

Source: AlSa'deh, Meinel, "Secure Neighbor Discovery: Review, Challenges, Perspectives, and Recommendations," *IEEE Security Privacy*, vol. 10, no. 4, pp. 26–34, 2012.

Modifier

- Used to implement hash extension
  - Higher protection against brute force attacks
  - Used for Sec>0
- Random Modifier
  - Same public key can generate multiple addresses
  - Addresses generated from same public key not linkable to each other
  - But: does not influence binding address to key
  - Non-crypto RNG may be used
- No subnet prefix in hash2
  - Final modifier can be reused if subnet prefix changes (e.g. mobile devices)
  - ➔ no new hash2 calculation in sender if only prefix changes ➔ less resources required

Privacy

- User Profiling?
  - If CGA is not changed ➔ user profiling possible
  - Recalculation of CGA requires effort
  - But possible to change CGA from time to time
  - Using same public key but different parameters
- Linking addresses from same public key?
  - Modifier should be random
  - Addresses generated from same public key unlinkable to each other
- Tracking mobile nodes based on CGA?
  - No, subnet prefix changes ➔ CGA changes

Solution for Mobile Devices?

- CGA creation
  - Huge computation effort
- If mobile node changes location and gets new address
  - Subnet prefix changes
  - Recalculation of CGA required
- Subnet prefix not in hash2 calculation
  - At least modifier can be re-used
  - Huge resource savings if Sec>0

CGA Security Considerations

- SHA-1?
  - Other hash functions allowed ➔ RFC4982
- If node is compromised
  - No need to fake address ➔ CGAs don't help
- Attacker could take someone else's CGA
  - Present it as non-cryptographical address (e.g. in DAD)
  - No way to distinguish
  - But SEND nodes prioritize information in signed messages
- Ideal: all nodes use SEND
  - Only CGAs and signed messages expected

Further SEND Options: Nonce and Timestamp

- Nonce: Number used only once
  - Used in Neighbor or Router Solicitation
  - Neighbor or Router Advertisement needs to include same nonce

  ➔ Prevents Replay Attack for solicited messages

- Timestamp
  - Put in unsolicited messages
  - Prevents replay attacks if clocks in nodes synchronized

  ➔ Prevents Replay Attack for unsolicited messages

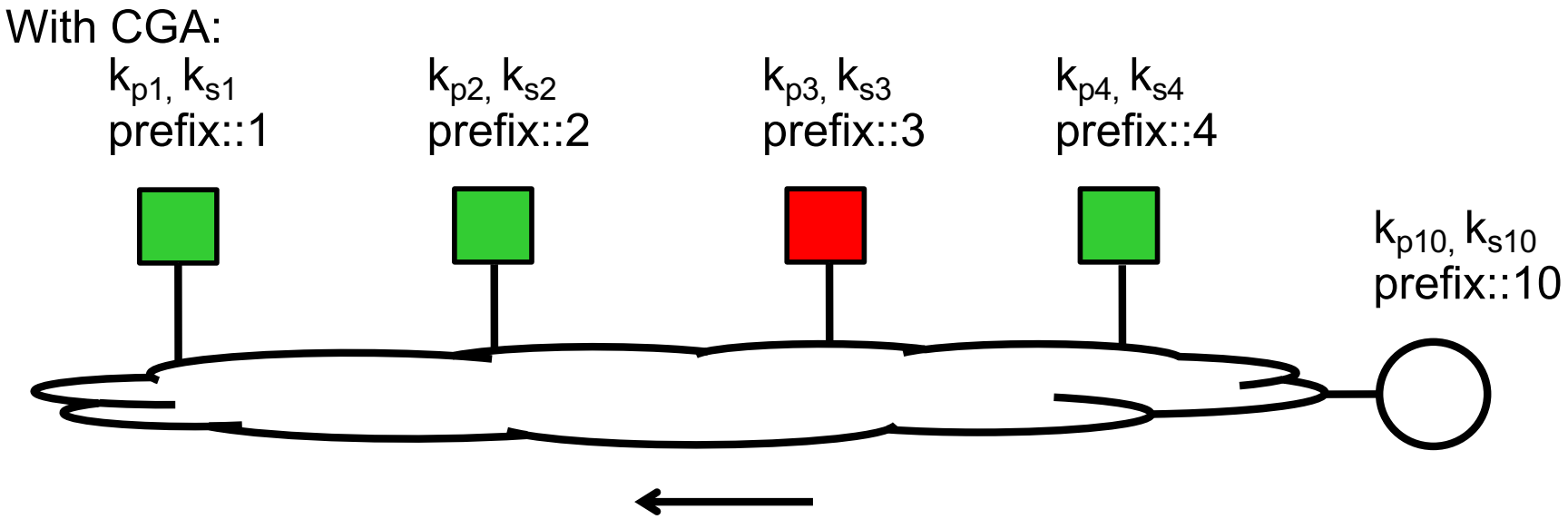
Router Authorization

Malicious Router

- Malicious node pretends to be router
  - Spoof Routing Advertisements
  - Injects wrong information

- Multiple methods to attack
  - Send wrong address prefix
  - Claim to be last hop router
  - Redirect traffic
  - Eavesdrop on traffic

CGA to prevent wrong Router Announcements?



not prevented with CGA as long as address ok

➔ Router Authorization

Router Authorization

- Authorize routers
  - to act as default gateways
  - to announce specifies IPv6 prefixes
- Requires ***Certificate*** from trusted third party
  - Node needs to have trust anchor (Certificate Authority)

- Two new ICMPv6 messages
  - *Certificate path solicitation (CPS)*
  - *Certificate path advertisement (CPA)*

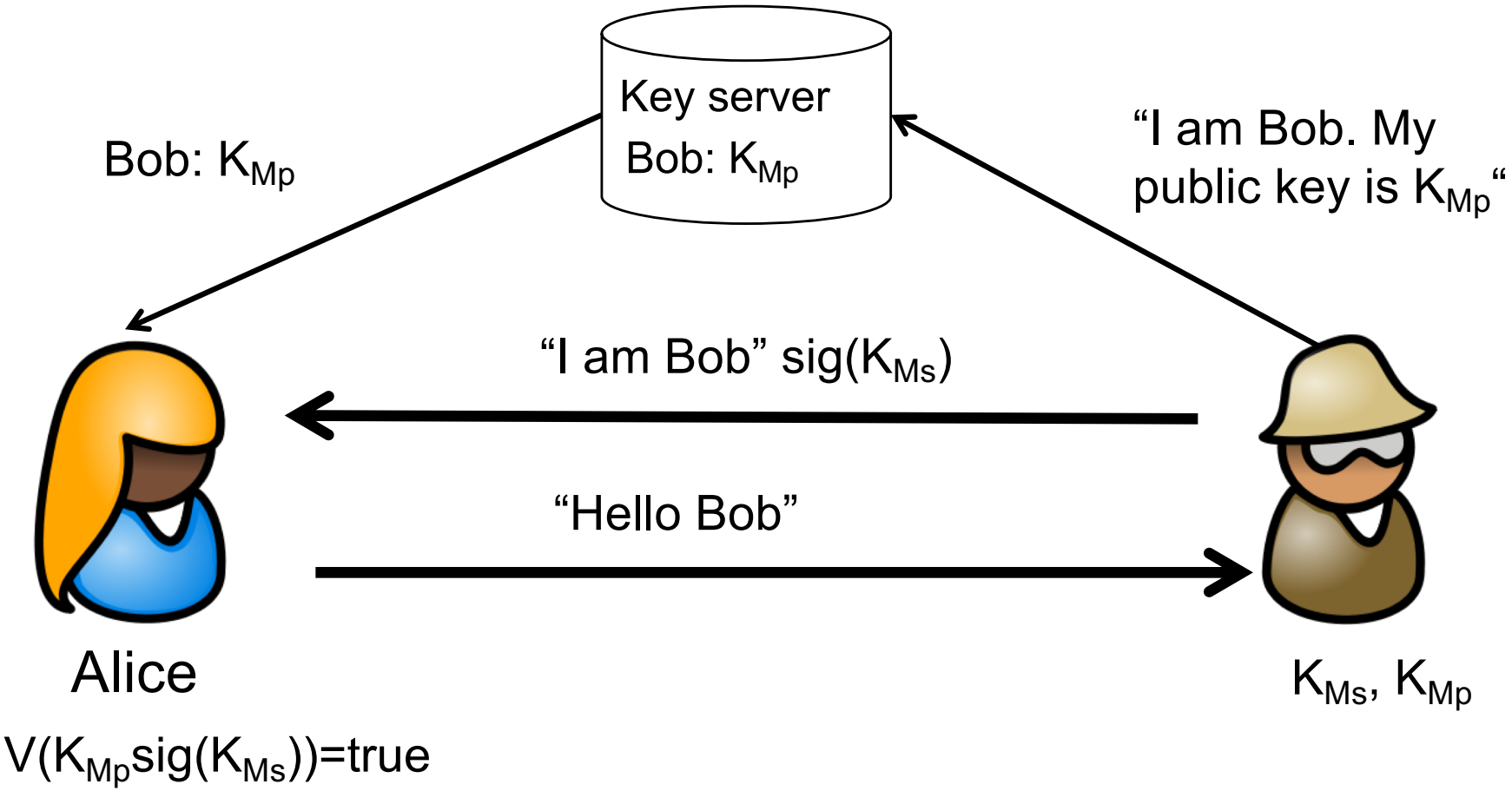
Recap: Certificates

- Public Key Cryptography
  - Based on public and private key pair
  - Each node keeps own private key

- Public keys need to be announced to the public

- What can happen?

Recap: Certificates



- Mallory announces k_{Mp} under the name Bob
- ➔ Mallory can impersonate Bob

Binding a Key to an Identity

- How to ensure that a public key really belongs to Bob?
- ➔ Binding public key to identity ➔ certificates

- Certificate Authority (CA)
  - Issues certificates
  - Identity - public key binding signed by CA
  - All nodes know public key of CA (e.g. preconfigured) ➔ can verify CA signature

Certificates

- CA checks Alice identity
  - Ensures that pubilc key belongs to Alice
  - Generates certificate for public key
- Certificate contains
  - Name
  - Public key
  - Signature from certificate authority (CA)
    - Hash over name and key signed with CAs private key

    Certificate=(“Alice”, k_{Ap}, sig_{Cs})
- Anyone can check signature (using CAs public key)
- ➔ CA ensures that public key belongs to Alice
  - only Alice has the matching private key

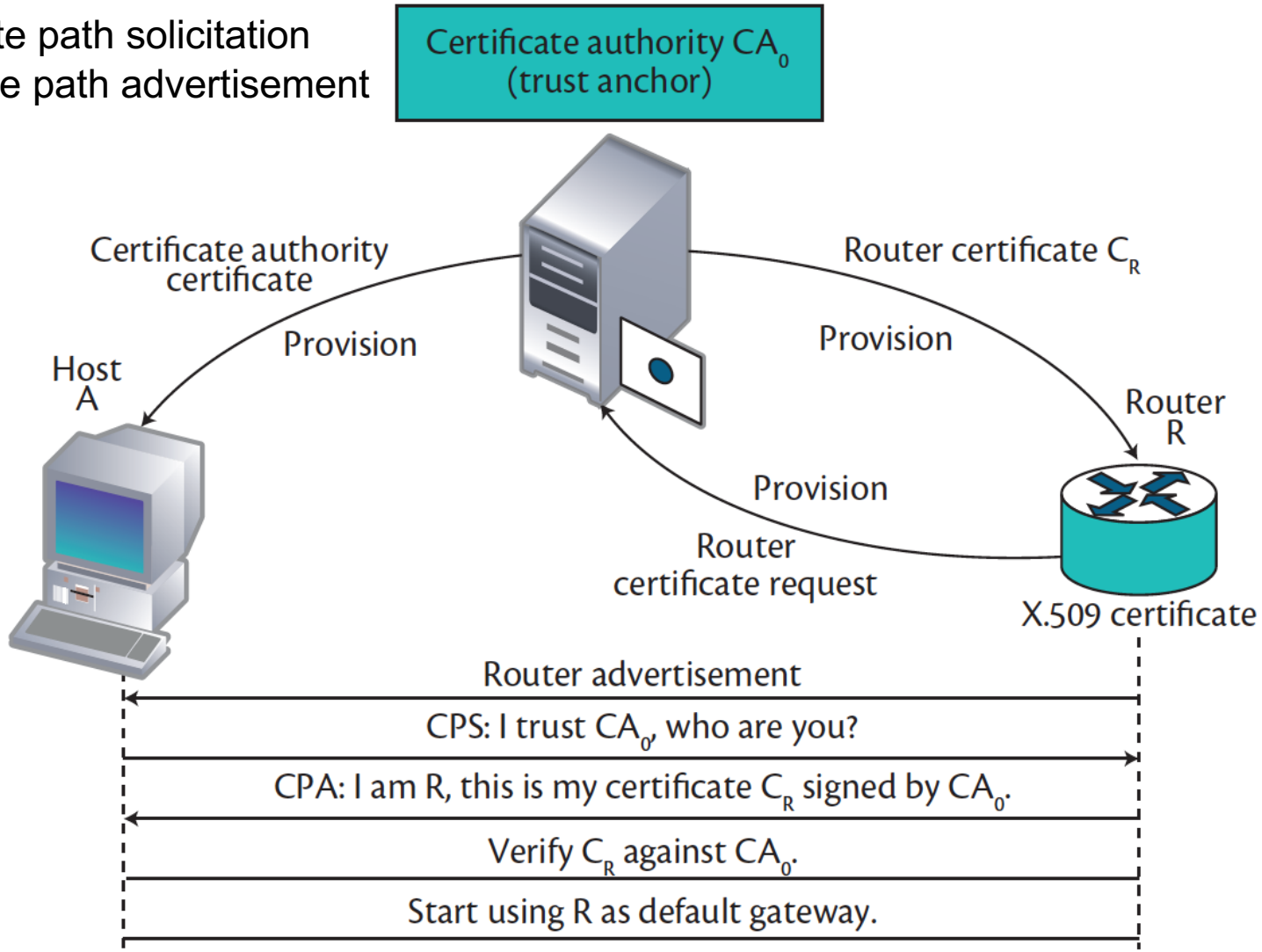
Public Key Infrastructures (PKI)

- PKI
  - Infrastructure and procedures to use public keys
  - System of Certificate Authorities (CAs)

- Functions
  - Key management
    - Key generation
    - Key distribution
  - Certificate management
    - Certificate creation
    - Certificate revocation

Router Authorization

CPS - Certificate path solicitation
CPA - Certificate path advertisement



Source: AlSa'deh, Meinel, "Secure Neighbor Discovery: Review, Challenges, Perspectives, and Recommendations," *IEEE Security Privacy*, vol. 10, no. 4, pp. 26–34, 2012.

SEND Drawbacks

- Resource consumption
  - Processing of CGAs and signatures ➔ computing
  - Overhead in messages ➔ bandwidth
  - Validation of certificates for routers
- PKI for router authorization

- Limitations
  - Attacks with non-CGA possible ➔ all nodes should use SEND
  - No confidentiality

- Rare Deployment
  - Few vendors support it

SEND Improvements

- CGA++ ➔ higher security
  - Includes subnet mask in hash2
- Using ECC instead of RSA
  - Shorter keys
- Delegate CGA calculation
  - Powerful server calculates CGA
- Monitoring the network
  - Detect suspicious behavior

SEND Summary

- Secure Neighbor Discovery
  - ICMPv6 Messages
- CGAs
  - Bind public key to IPv6 address
  - Prevent address theft
  - Computational Effort
- Router Authorization
  - Authorize nodes to act as router
  - Using Certificates and Trust Anchor (Certificate Authority)

Thank you!