802.11 Denial-of-Service Attacks: Real Vulnerabilities and Practical Solutions

John Bellardo and Stefan Savage, Department of Computer Science and Engineering

— The Deauthentication Attack Explained—

The 802.11 standard requires all client nodes in a network to associate with an access point before transmitting data. Association involves two state transitions, pictured to the right. In addition to the forward transitions there is a backward transition, call deauthentication.

The deauthentication transition is not verified in any way, shape, or form. The access point that receives the request blindly honors it, leaving that transition ripe for exploitation by an attacker. The following series of figures shows how an attacker can take advantage of this situation to mount a denial-of-service attack on 802.11 infrastructure wireless networks.

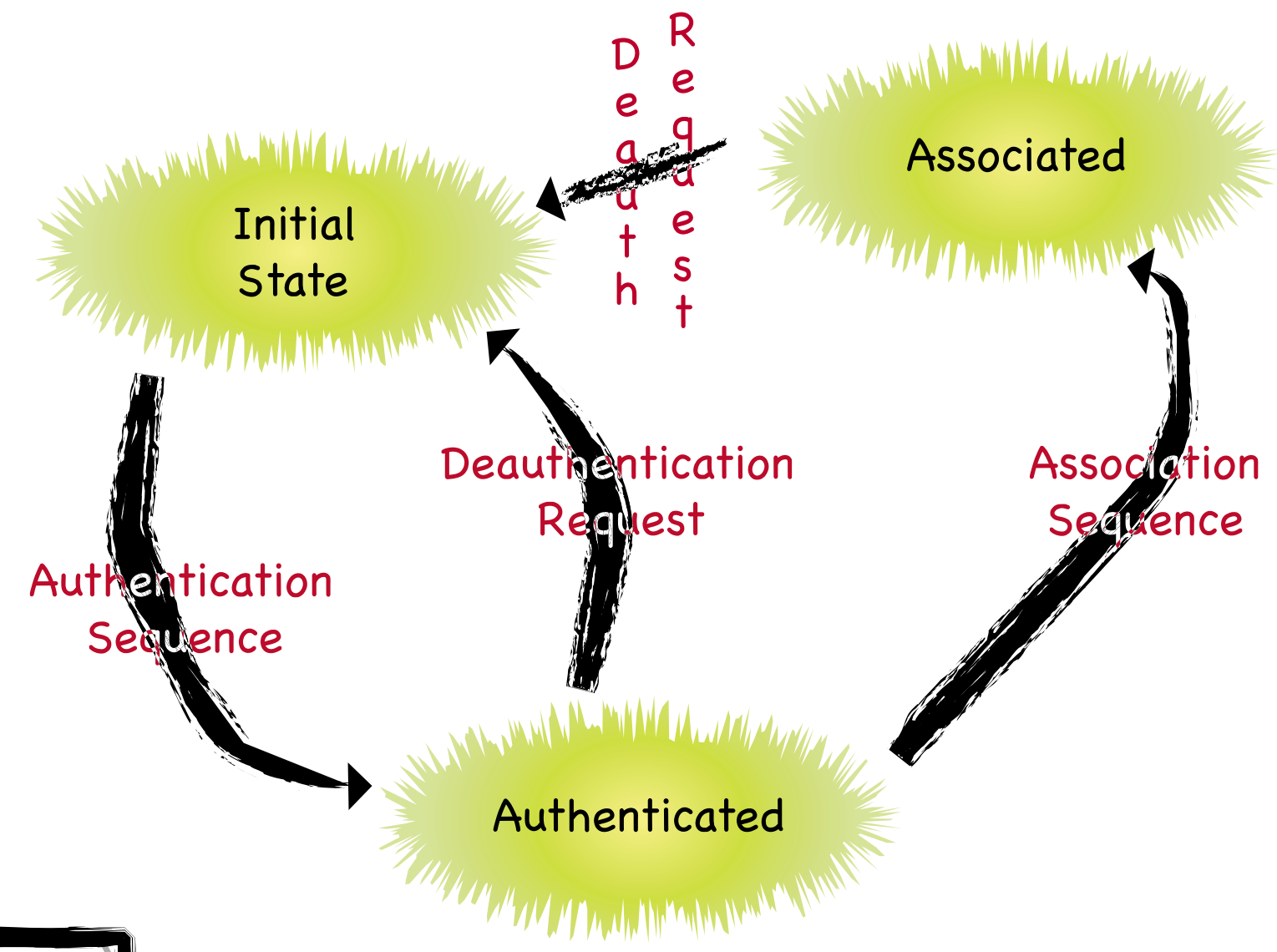
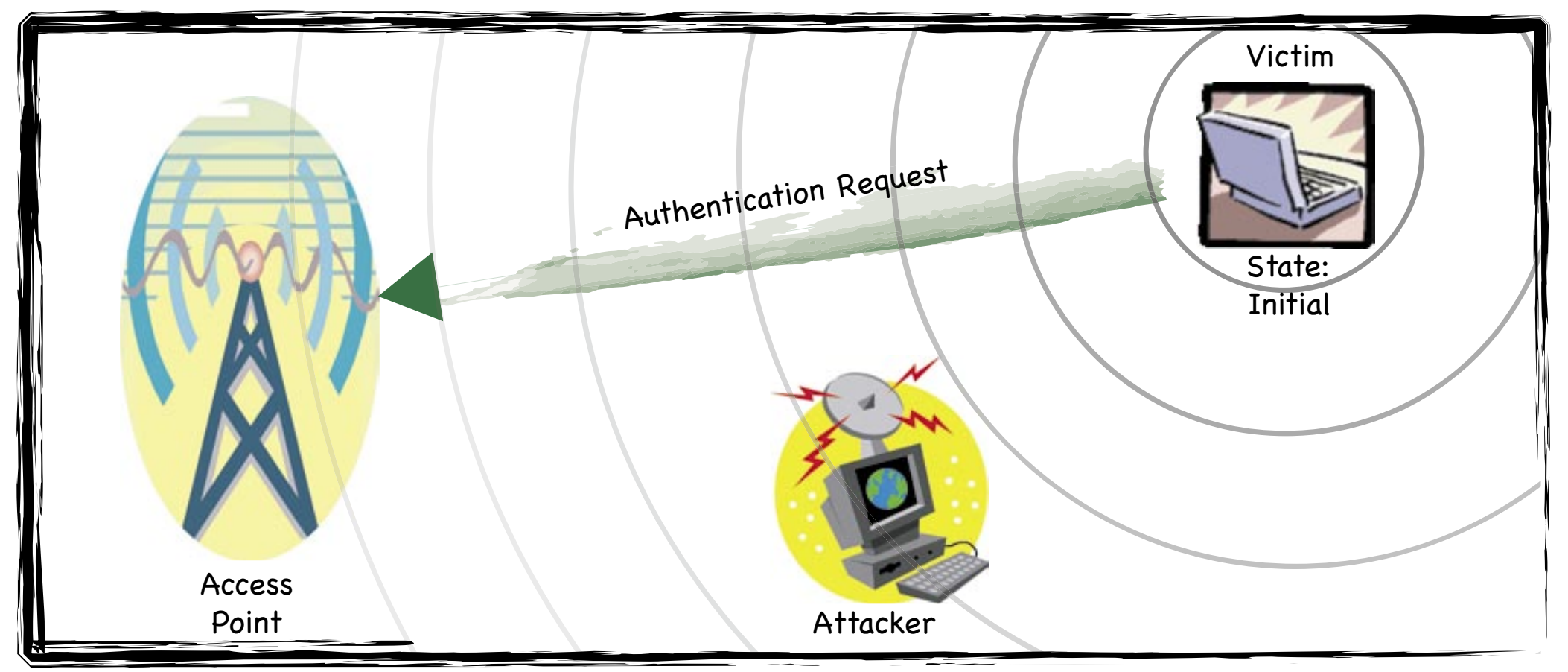
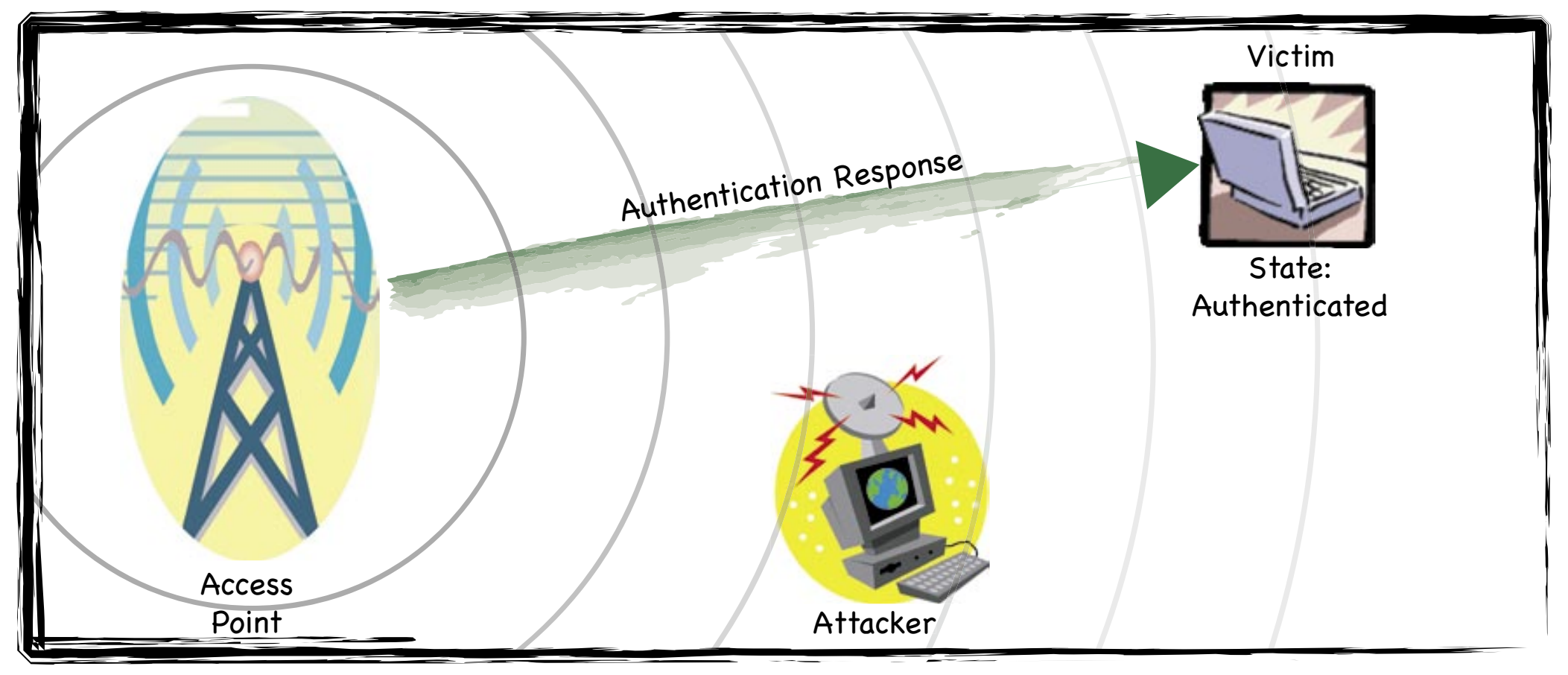


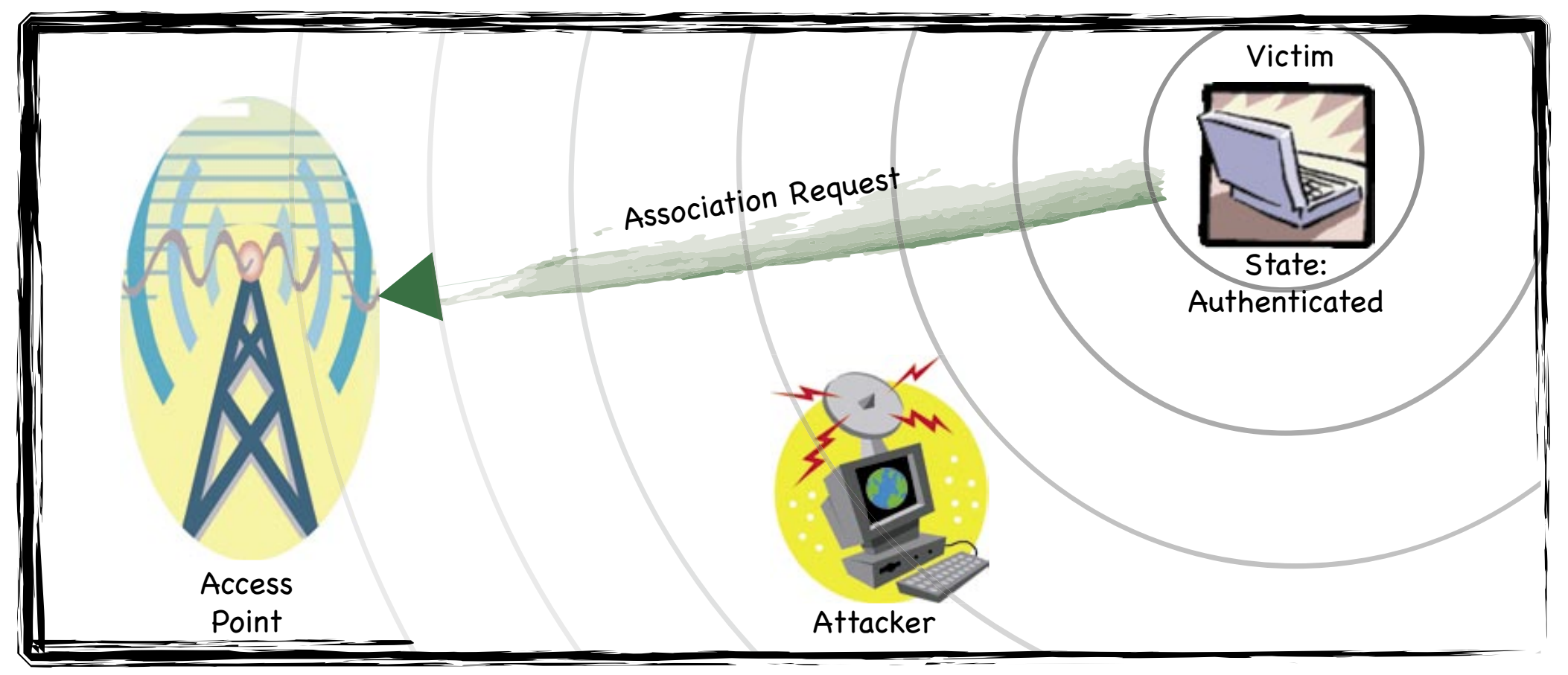
Figure 1:
Nodes must be associated before sending data, which requires the transitions depicted above.



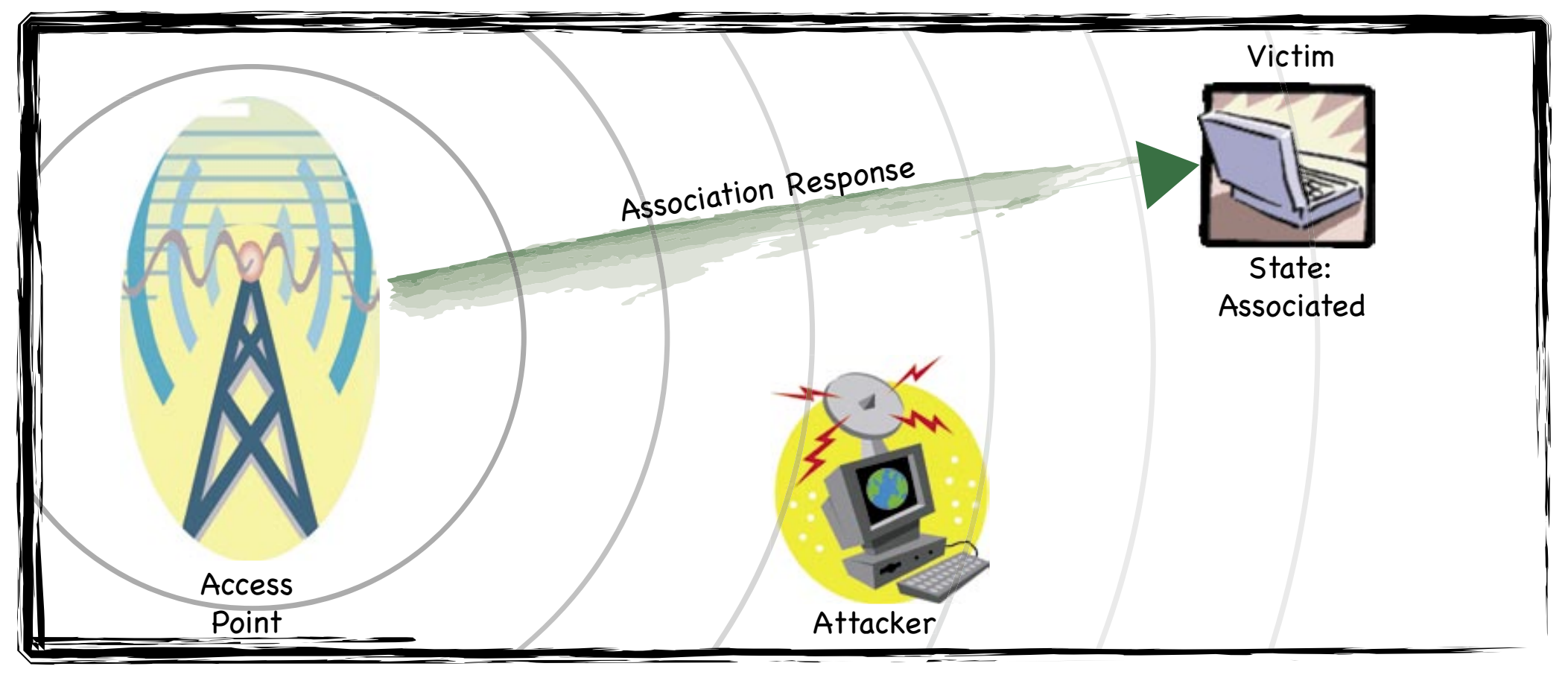
Step 1: The victim initiates authentication with the access point. The attacker stalks his prey.



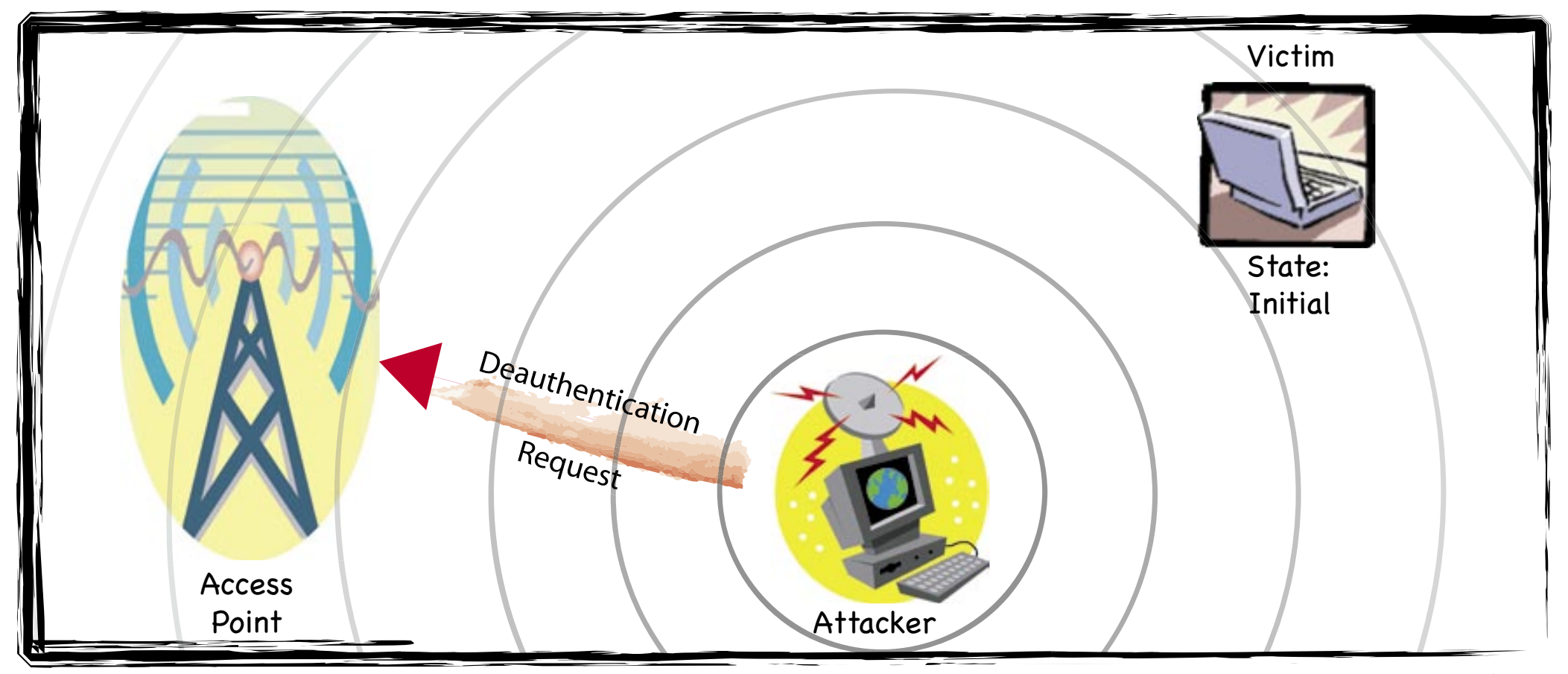
Step 2: The victim completes authentication with the access point.
The attacker continues stalking.



Step 3: The victim initiates association with the access point. The attacker still stalks.



Step 4: Association completes. The victim is now ready to send data, but...



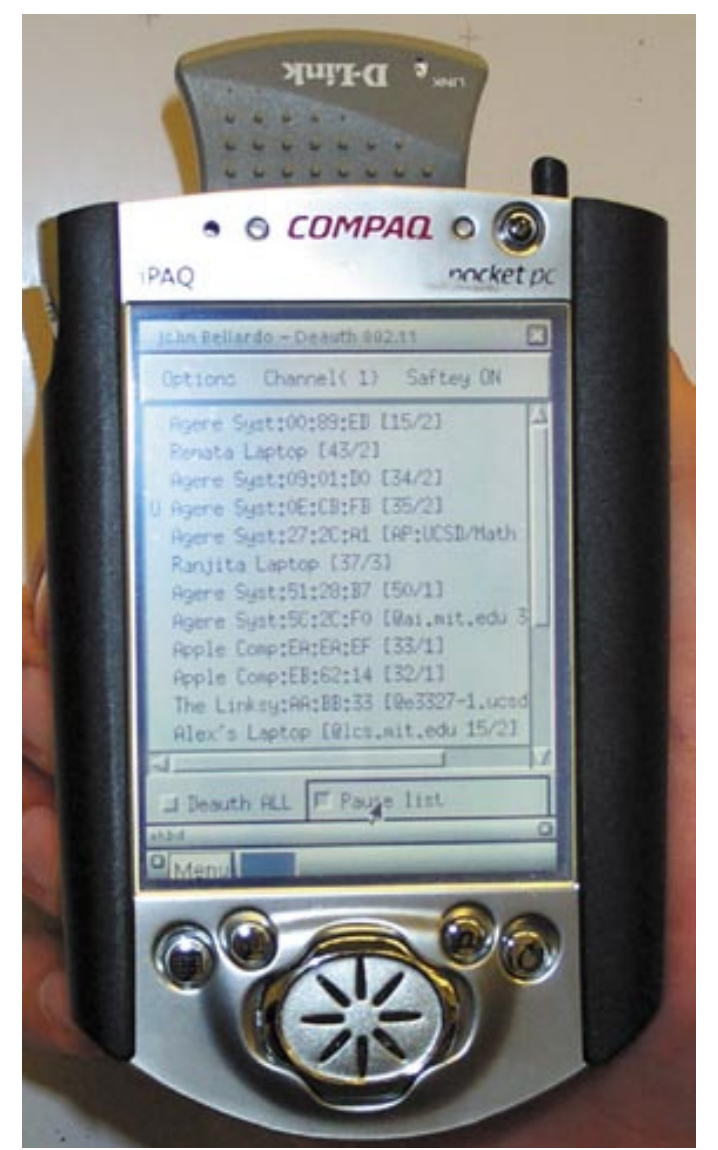
Step 5: The attacker quickly pounces by sending a deauthentication request on "behalf" of the victim, forcing the victim to revert to the initial state unable to send data.

— Abstract —

The combination of free spectrum, efficient channel coding and cheap interface hardware have made 802.11-based access networks extremely popular. For a couple hundred dollars a user can buy an 802.11 access point that seamlessly extends their existing network connectivity for almost 100 meters. As a result, the market for 802.11-based LANs exceeded $1 Billion in 2001 and includes widespread use in the home, enterprise and government / military sectors, as well as an emerging market in public area wireless networks. However, this same widespread deployment makes 802.11-based networks an attractive target for potential attackers. Indeed, recent research has demonstrated basic flaws in 802.11's encryption mechanisms and authentication protocols — ultimately leading to the creation of a series of protocol extensions and replacements (e.g., WPA, 802.11i, 802.1X) to address these problems. However, most of the previous work has focused *primarily* on the requirements of access control and confidentiality, rather than availability.

In contrast, this work focuses on the threats posed by denial-of-service (DoS) attacks against 802.11's MAC protocol. Such attacks, which prevent legitimate users from accessing the network, are a vexing problem in all networks, but they are particularly threatening in the wireless context. Without a physical infrastructure, an attacker is afforded considerable flexibility in deciding where and when to attack, as well as enhanced anonymity due to the difficulty in locating the source of individual wireless transmissions. Moreover, the relative immaturity of 802.11-based network management tools makes it unlikely that a well-planned attack will be quickly diagnosed.

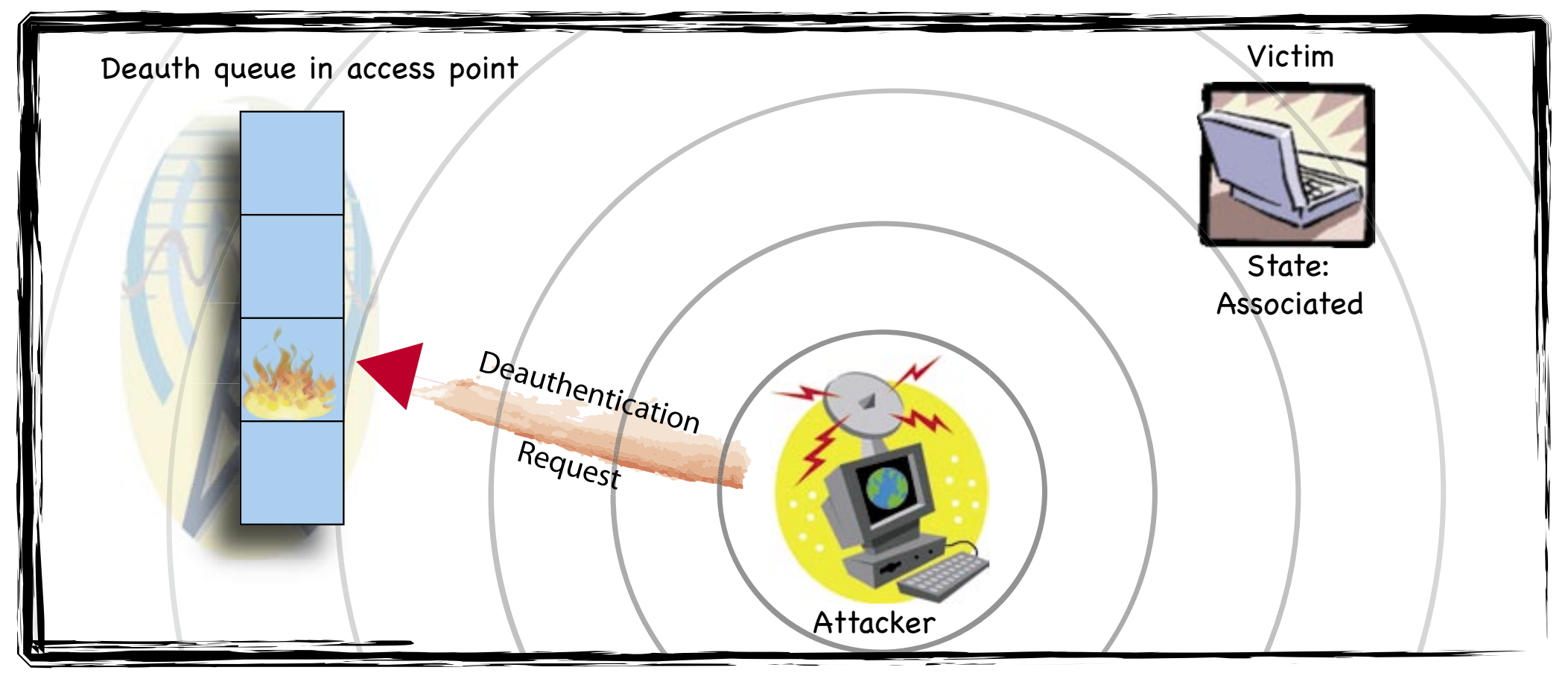
This poster explains three of the principal contributions of this work. First, we provide an overview of vulnerabilities in the 802.11 management and media access services that are vulnerable to attack. Second, we implement two important classes of denial-of-service attacks and investigate the range of their practical effectiveness. Finally, we describe, implement, and evaluate non-cryptographic countermeasures that can be implemented in the firmware of existing MAC hardware.



Our Wireless Attack Tool

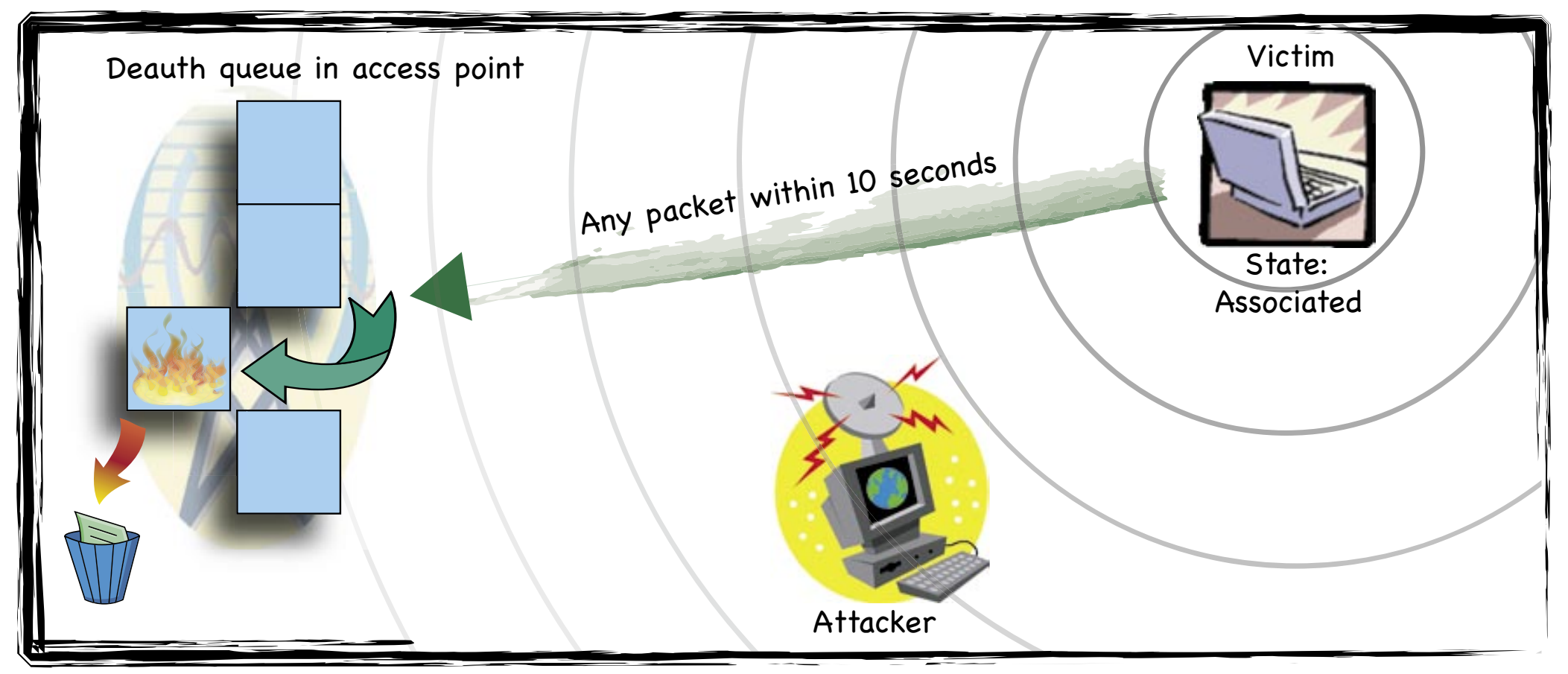
—Defending against the Deauthentication Attack—

Our proposed defense against the deauthentication attack is straightforward. An access point, upon receiving a deauthentication request, places it on a wait queue for a certain period of time. If time expires and no other traffic from that node has been seen, the request is honored and the node deauthenticated. On the other hand, if traffic from that node is seen before time expires, the request is dropped and *not* honored.

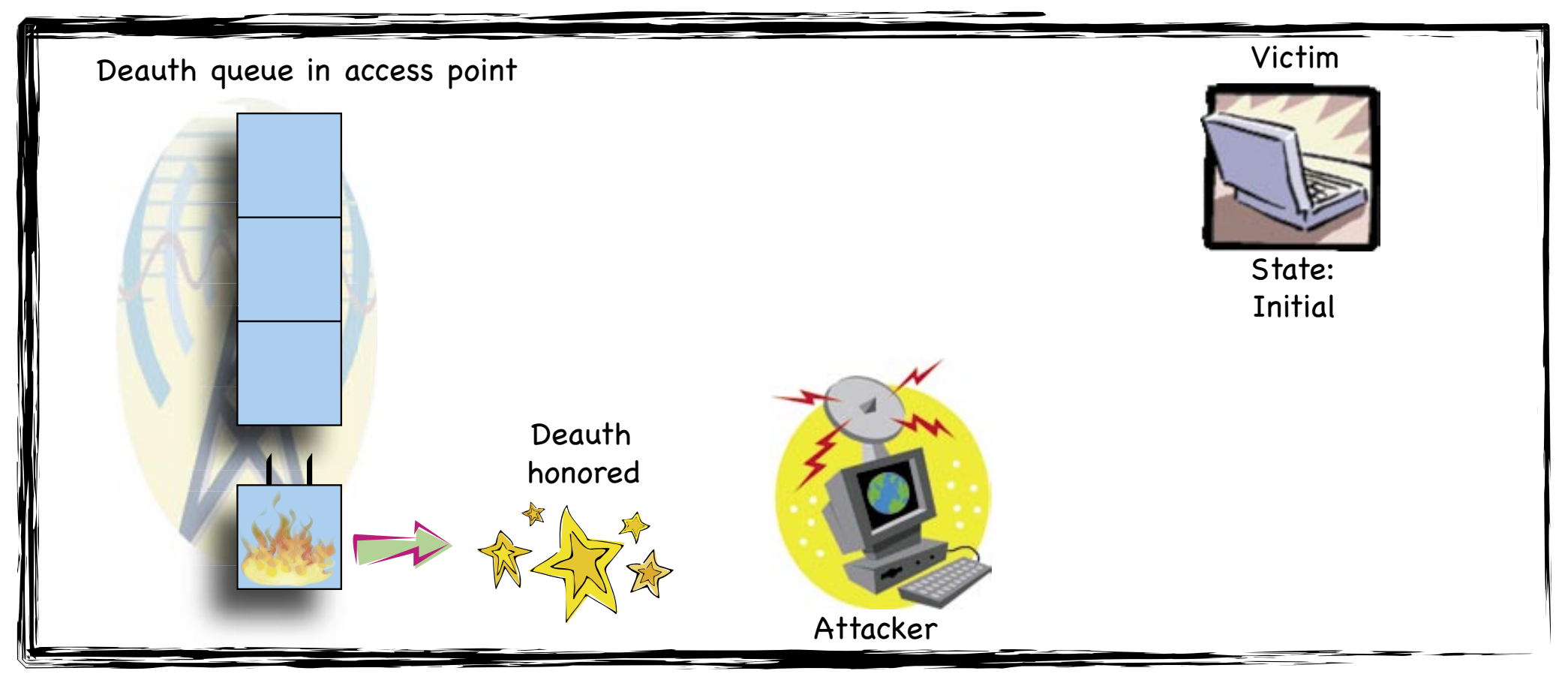


Defense Step 1: A deauthentication packet is received, either from the attacker or the victim, and is placed in a wait queue.

| OR |



Defense Step 2a: If a legitimate packet is received from the victim the deauthentication packet is discarded.



Defense Step 2b: If no legitimate packet is received from the victim before time expires, the deauthentication is honored.

—Evaluating the Deauthentication Attack—

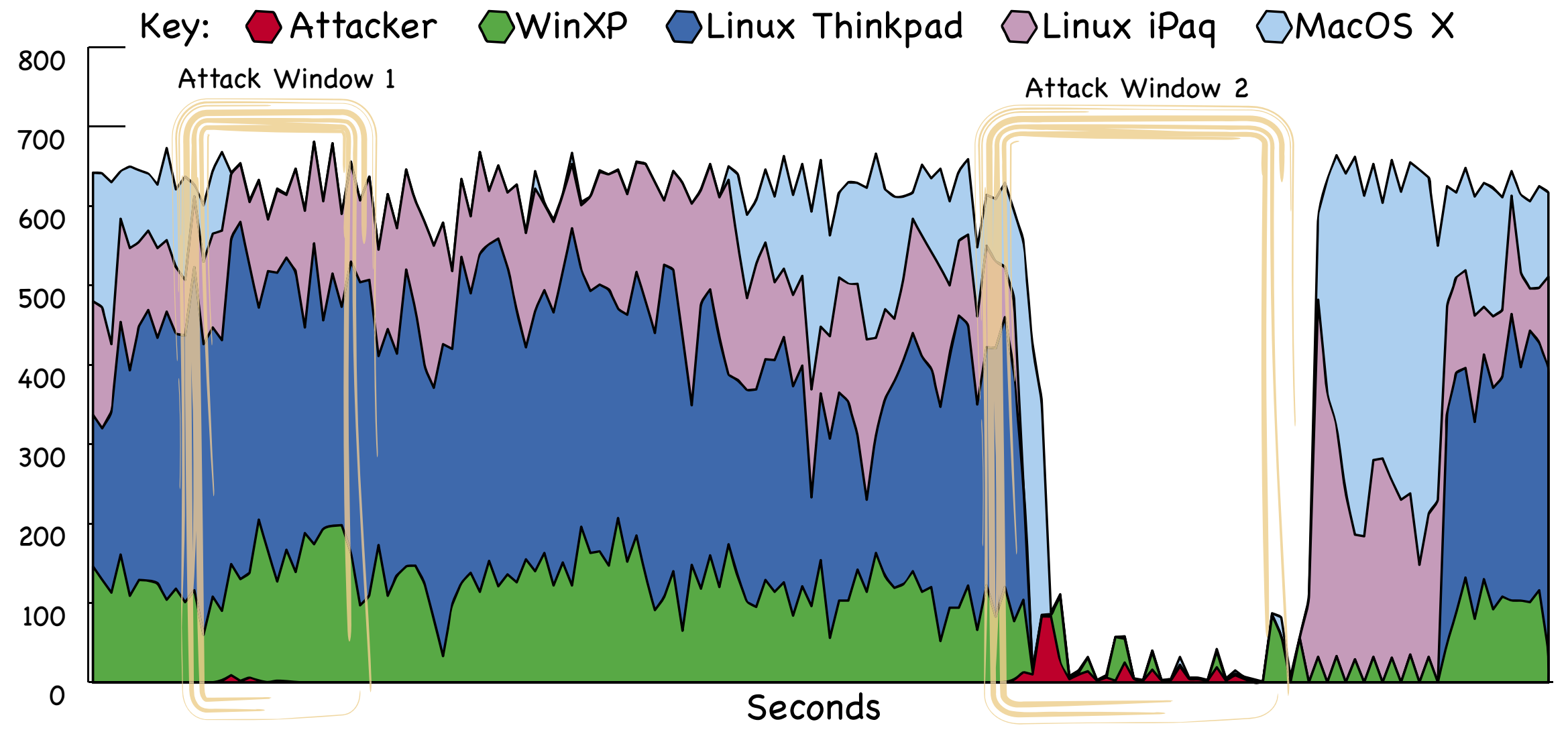
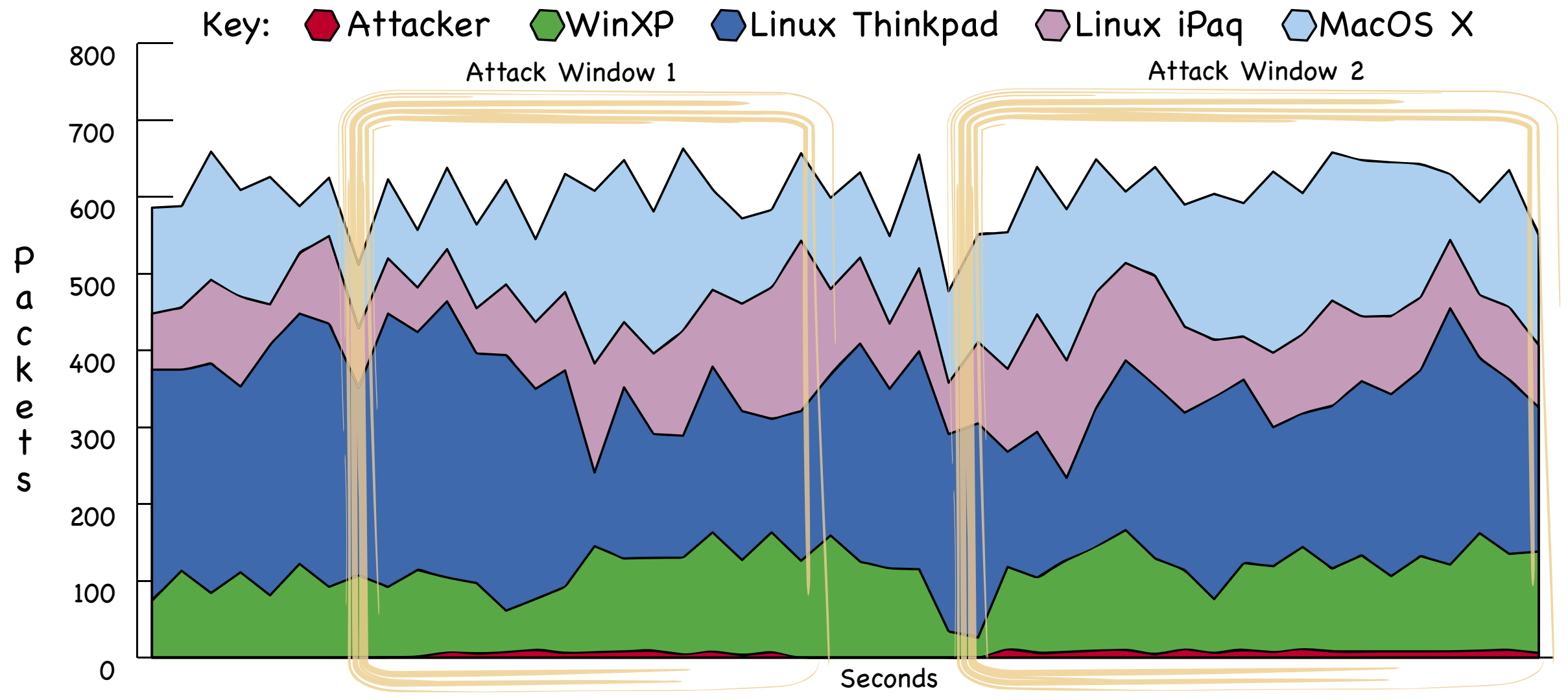


Figure 2, above: Evaluating the Deauth Attack

The above graph depicts two actual deauthentication attacks. In the first attack only a single machine was targeted, the MacOS X machine. The attacker was able to completely stop that machine from generating network traffic without adversely effecting the other machines using the network. In the second attack interval all machines on the network were attacked. Virtually all of the traffic from those nodes was stopped, as seen in the graph. The WinXP machine was able to transmit a few packets due to a race condition in our implementation of the attack, but these packets weren't accomplishing any actual communication. Even though this graph shows just a single experiment, we have performed many other tests of the deauthentication attack. In all cases the attack has been successful at shutting down the victim, or in some cases the set of victims, targeted.

The graph below shows our evaluation of the deauth defense. We hardened an access point in the lab and duplicated the experiments we performed to get figure 2. In both instances the attack was unsuccessful, demonstrating the viability of our defense.

Figure 3, below: Evaluating the Deauth Defense



—The NAV Attack Explained—

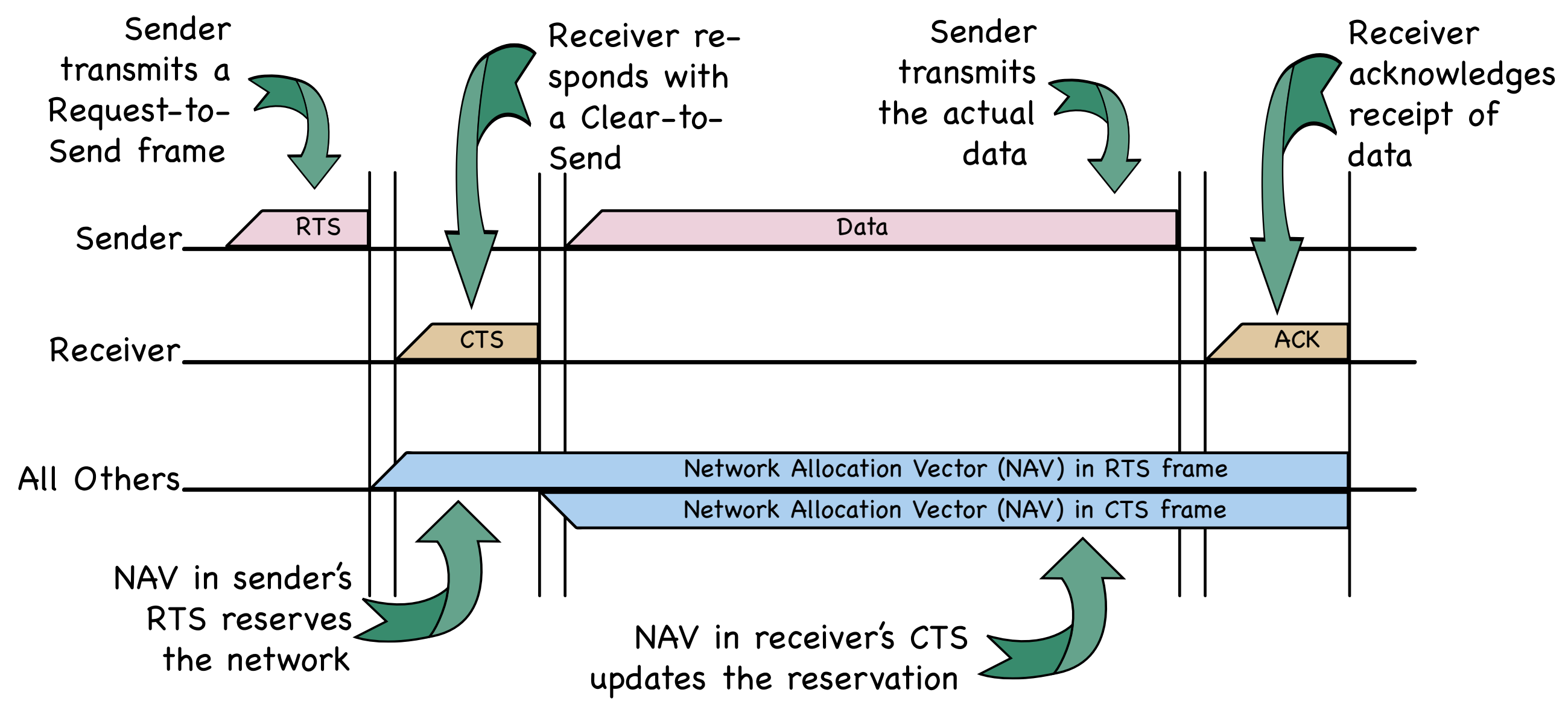

Sender transmits a Request-to-Send frame
Receiver re-sponds with a Clear-to-Send
Sender transmits the actual data
Receiver acknowledges receipt of data
Sender
Receiver
All Others
RTS
Data
CTS
ACK
Network Allocation Vector (NAV) in RTS frame
Network Allocation Vector (NAV) in CTS frame
NAV in sender's RTS reserves the network
NAV in receiver's CTS updates the reservation

Figure 4: A time line showing how NAV is legitimately used to implement the RTS/CTS functionality.

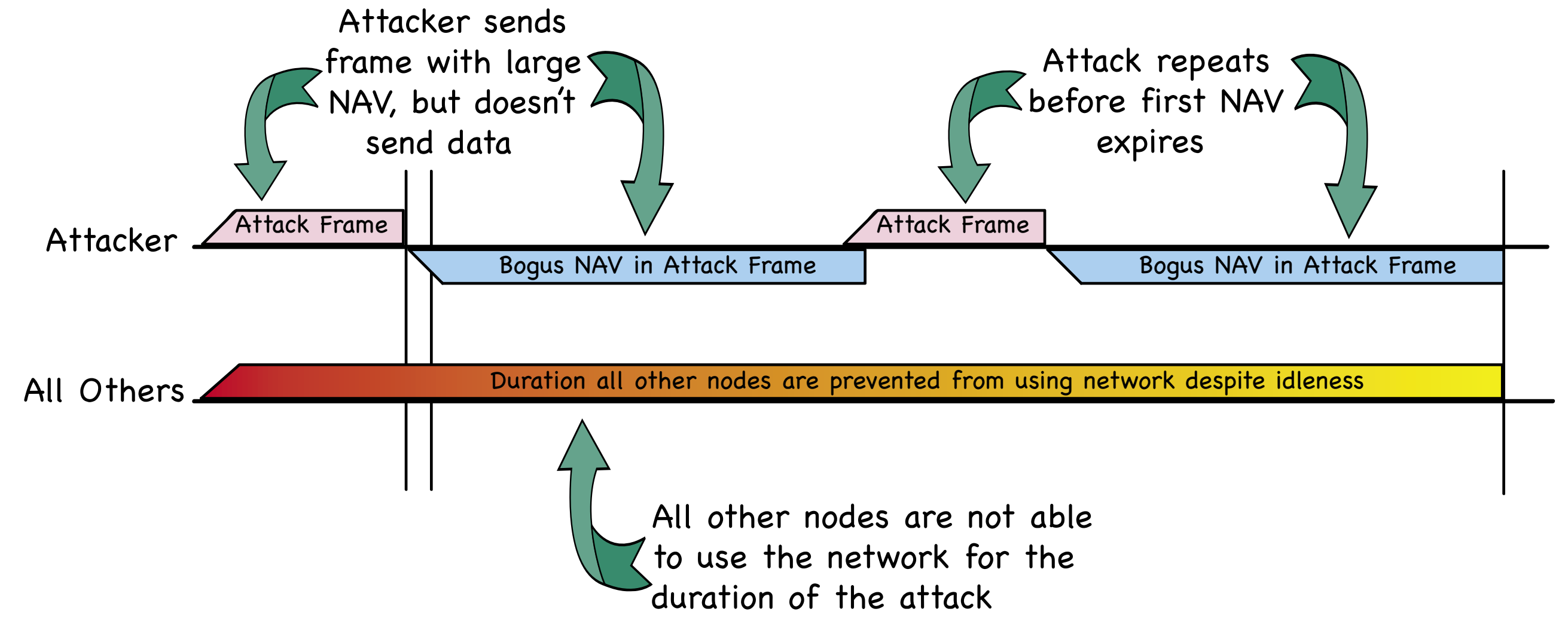

Attacker sends frame with large NAV, but doesn't send data
Attack repeats before first NAV expires
Attacker
All Others
Attack Frame
Bogus NAV in Attack Frame
Attack Frame
Bogus NAV in Attack Frame
Duration all other nodes are prevented from using network despite idleness
All other nodes are not able to use the network for the duration of the attack

Figure 5: Another time line showing how NAV can be exploited to DoS the wireless network.

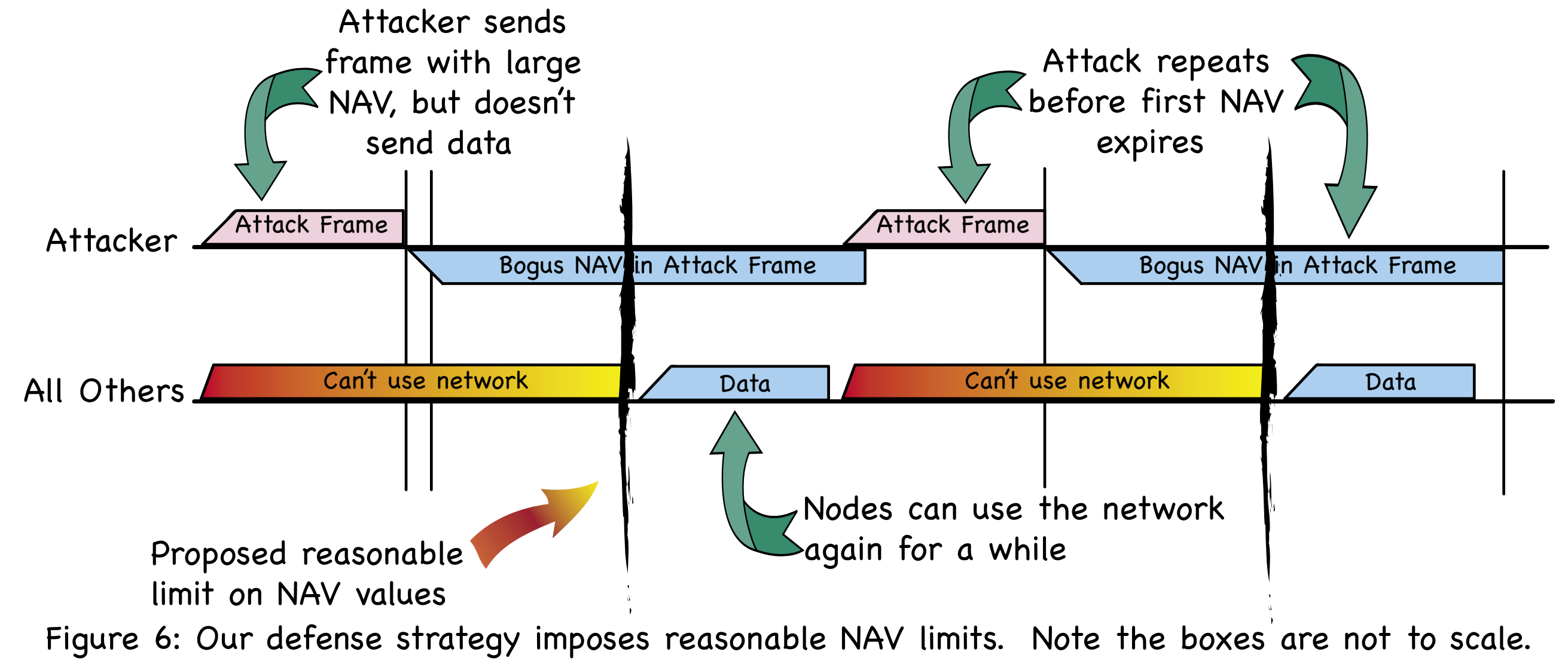

Attacker sends frame with large NAV, but doesn't send data
Attack repeats before first NAV expires
Attacker
All Others
Attack Frame
Bogus NAV in Attack Frame
Can't use network
Data
Proposed reasonable limit on NAV values
Nodes can use the network again for a while

Figure 6: Our defense strategy imposes reasonable NAV limits. Note the boxes are not to scale.

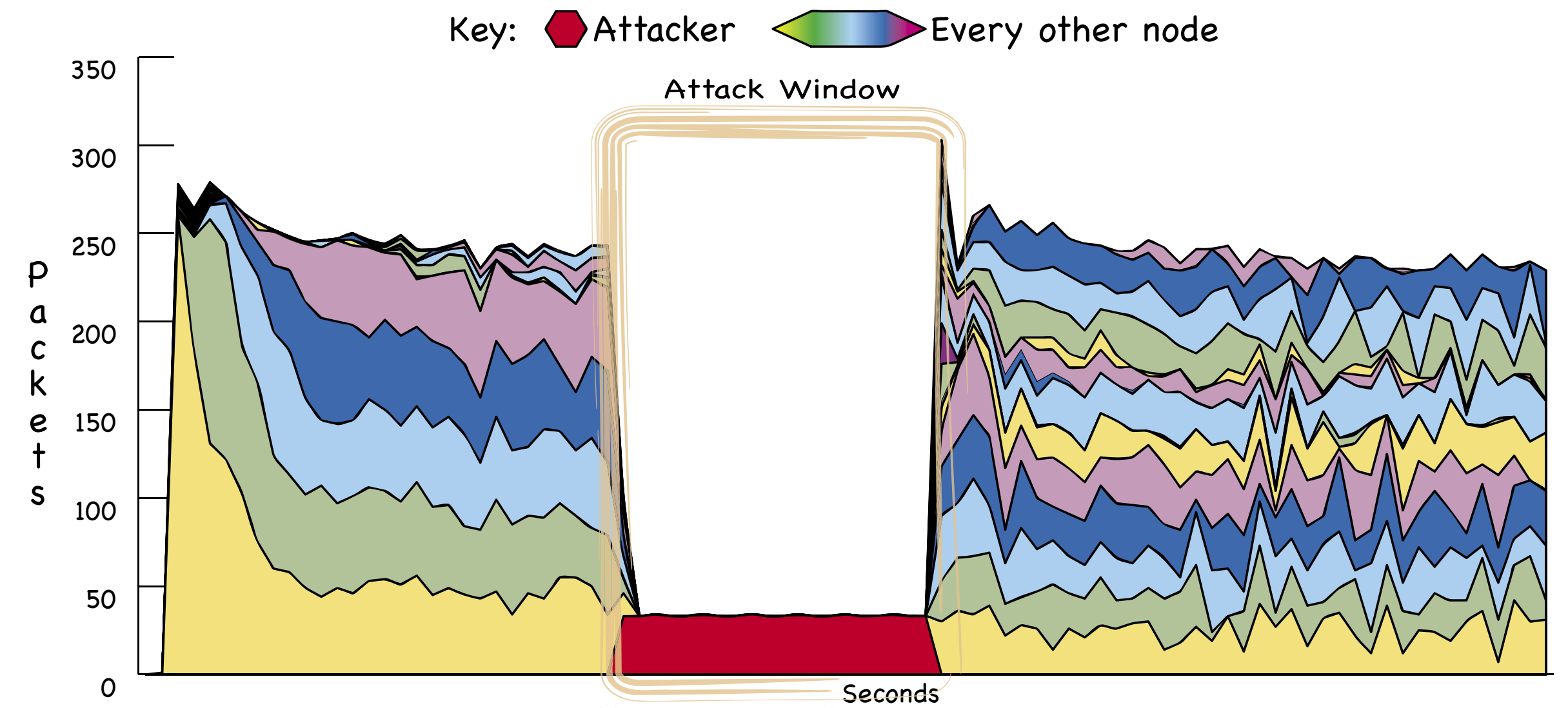


Figure 7: A graph show the effectiveness of the NAV attack, as simulated by NS2.

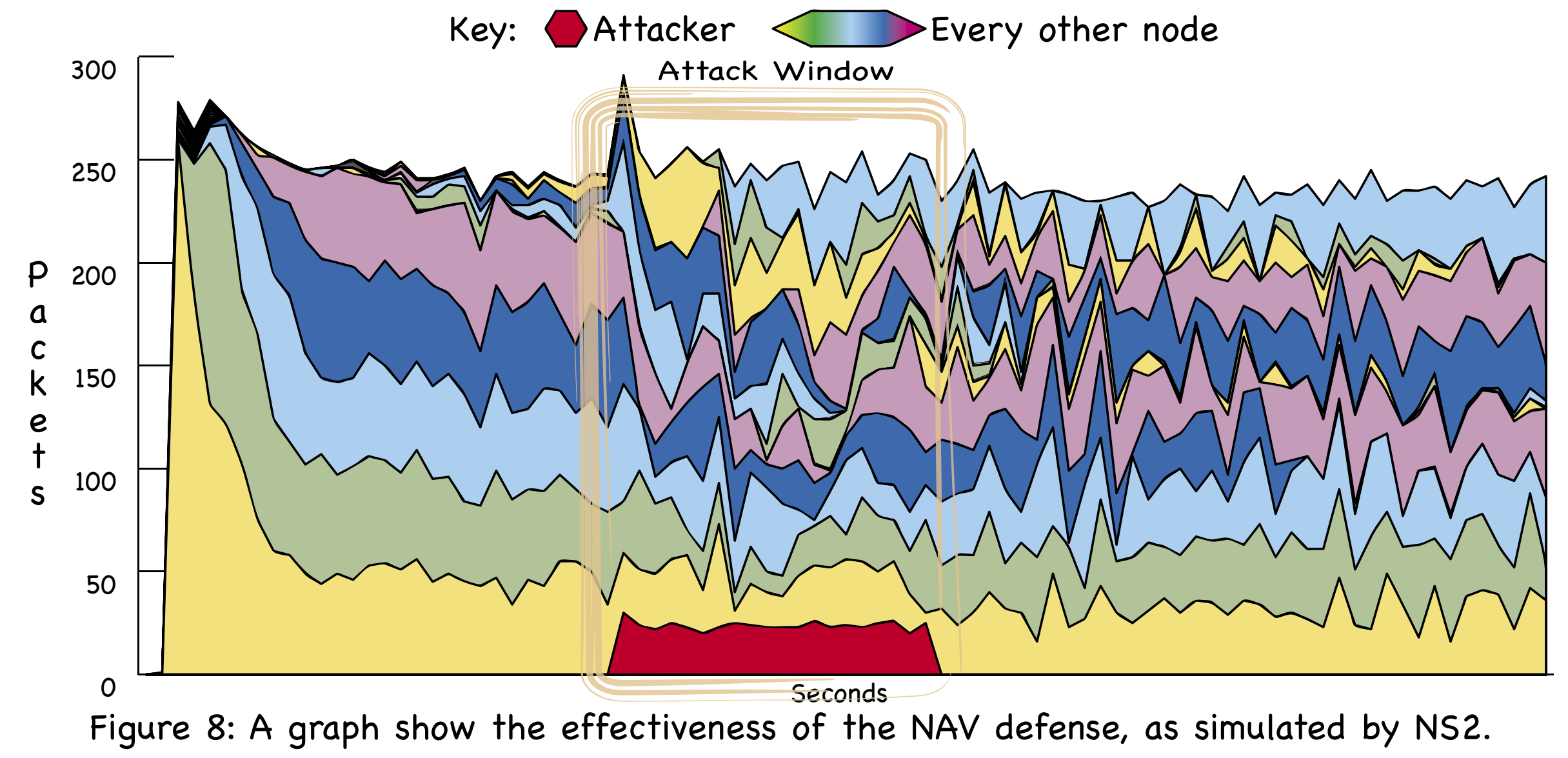


Figure 8: A graph show the effectiveness of the NAV defense, as simulated by NS2.

In Proceedings of the USENIX Security Symposium, Washington D.C., August 2003. The full paper and more information is available at http://ramp.ucsd.edu/~bellardo