The beacons stuffing is based on the “push model” of information delivery. The key idea is to overload IEEE 802.11 beacons to carry additional information. Beacon frames are a local (100-200 meters if it is not changed by transmitting power or encoding scheme) broadcast that are used to announce the presence of a Wi-Fi network.

We treat the information to be broadcast as a string of bytes. In most cases, we expect the information to be a short text message. However, with this techniques we can also send files.

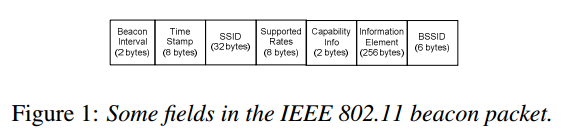
The AP splits the message into smaller fragments, and transmits each fragment in a separate beacon. The size of the fragment depends on the mechanism being used. The fragment sent in each beacon has the following format:

UniqueID : SequenceNumber : MoreFlag : InfoChunk

UniqueID identifies the message being broadcast, the SequenceNumber is the fragment number, and MoreFlag informs the client if it should expect more fragments. Finally, the InfoChunk has the contents of the message. Clients reassemble the message after receiving all fragments.

We now discuss details of three specific techniques to carry the messages in a beacon – SSID, BSSID and Beacon Information Element.

The main difference between the techniques is the field in the beacon packet that is used to carry the messages. None of the three techniques require any modifications to the hardware or firmware of the client device to receive the messages. For the SSID and BSSID based techniques, a simple user-level application is sufficient to reassemble the fragmented messages. The third technique, which uses Information Element, requires changes to the Wi-Fi driver on the client devices.



**SSID Concatenation:** The SSID field in the Beacon carries the name of the wireless network. The maximum length is 32 bytes. Assuming the UniqueID is 1 byte and SequenceNumber and MoreFlag can fit in 1 byte, we are left with 29 bytes for the InfoChunk. Fragments are transmitted in successive beacons. The maximum length of each unique message is 3712 bytes.

This approach is easy to implement with the user interface and to set the SSID. A simple user-level program is sufficient reassemble the fragments and display the reassembled message.

This approach enables to send messages at the rate of 23Kbps with a 10ms AP Beacon Interval. However, this approach has a significant limitation. Most client devices include an application that displays the SSIDs of networks within the range of the device. Unless this application is modified, the user interface of this application will get swamped with the large number of SSIDs, which might obscure legitimate SSIDs.

**BSSID Concatenation:** BSSIDs are 6 byte unique identifiers of an AP. Generally, they are set to be equal to the MAC address of the AP. However, these can generally be set to any value. Assuming once again that the UniqueID is 1 byte and SequenceNumber and MoreFlag can fit in 1 byte, we can transmit 4 bytes of the message in a beacon. This gives us a maximum length of 512 bytes for each unique message.

All beacons have their SSID set to a fixed, well-known SSID. When a client receives a beacon with this SSID, it queries the BSSID field from the user-level. It assembles the message after receiving all fragments.

This approach overcomes the primary limitation of the SSID concatenation. The user is only presented with a list of unique SSIDs, and when the user selects a particular SSID, the driver determines which BSSID to connect to. The only limitation of this approach is the bandwidth. Assuming that beacons are sent every 10ms, the transmission rate is 3.2Kbps.

**Beacon Information Element:** The IEEE 802.11 standard allows AP vendors to add 253 bytes of vendor-specific Beacon Information Elements in its beacon. We use this feature to define a special BIE for broadcasting information. Each message is fragmented into 251 byte chunks and sent in successive beacons. All beacons have their SSID set to a fixed, well-known SSID. The Wi-Fi driver at client devices are modified to recognize this BIE and pass it to the user level. This approach provides a bandwidth of approximately 200Kbps, which is higher than the other two approaches. It also does not spam the wireless UI of the client device. The main drawback of this approach is that it requires driver modification for client devices, making it relatively difficult to deploy.

For each of the above encoding schemes, messages that span multiple beacons run the risk that if any fragment is lost then the entire message is lost. One can combat this problem by using forward error correction when encoding the message.