

# Drückerei

## Den Amazon Dash Button zweckentfremden



**Der Dash Button ist ein Plastikschächtelchen, auf dem ein Markenname aufgedruckt ist. Sein einziger Zweck: Drückt man darauf, bestellt er via WLAN das Produkt bei Amazon. Mit einem Trick können Sie damit beliebige Aktionen auslösen.**

Von Johannes Merkert

**D**er Wecker klingelt. Sie drücken den Dash Button auf dem Nachttisch und der Wasserkocher geht an. Beim Frühstück drücken Sie auf einen anderen Knopf, der die Pkw-Standheizung anfeuert. Und beim Rausgehen betätigen Sie Knopf 3, der den Außer-Haus-Modus des Smarthome aktiviert. Knöpfe dieser Art haben viele Hersteller für teuer Geld im Portfolio. Sie kosten viel, weil die Knöpfe mit Microcontroller und Funk-Hardware ausgestattet sind. Amazons Dash Button

enthält vergleichbare Technik, kostet aber nur 5 Euro und der Kaufpreis wird mit der ersten Bestellung verrechnet.

Beim Druck auf den Knopf bootet die Firmware, der Button verbindet sich mit dem WLAN, kommuniziert mit Amazon-Servern und schaltet sich danach wieder aus. Diese kurze Wartezeit soll Strom sparen, denn die fest eingebaute AAA-Batterie reicht nur für rund 1000 Tastendrücke.

Eigentlich will Amazon nicht, dass man etwas anderes damit macht, als so das aufgedruckte Produkt zu bestellen. An zwei Stellen können Sie Amazon jedoch reingrätchen: Zum Ersten können Sie erkennen, dass der Button sich mit dem WLAN verbindet, und daraufhin eine beinahe beliebige Aktion auslösen. Zum Zweiten können Sie verhindern, dass die Daten, die der Button verschickt, die Amazon-Server erreichen.

### Grätsche links

Um die Verbindungsversuche zu erkennen, brauchen Sie einen Rechner im heimischen WLAN, der immer an ist. Der Rechner sollte wenig Strom verbrauchen und wenig kosten. Deshalb eignet sich dafür ein Raspi, idealerweise ein Raspi 3, der bereits eine WLAN-Schnittstelle mitbringt. Ein Raspi 1 oder 2 ginge aber auch, wenn Sie ein günstiges WLAN-Modul per USB anschließen.

Auf diesem Rechner läuft ein Skript, das permanent das WLAN-Interface beäugt, um zu erkennen, ob gerade ein Dash Button gebootet hat. Dann nämlich versendet er nach Erhalt seiner IP-Adresse einen ARP Request, und zwar an die MAC-Adresse 00:00:00:00:00:00, um festzustellen, ob die erhaltene IP-Adresse wirklich noch frei ist. Wenige Zeilen Python-Code filtern ein solches Paket aus dem Strom heraus:

```
from scapy.all import sniff, ARP
def arp_received(packet):
    if packet[ARP].op == 1 and \
        packet[ARP].hwdst == \
        '00:00:00:00:00:00' and \
        packet[ARP].hwsrc == \
        '50:f5:da:6f:98:6c':
        print("Button gedrückt!")
sniff(prn=arp_received,
      iface="wlan0", filter="arp")
```

Das Python-Modul `scapy` enthält unter anderem einen Netzwerk-Sniffer. Im Bei-

spiel belauscht er das Device „wlan0“ und ruft bei Eintreffen eines ARP-Pakets (filter=:arp:) die Funktion arp\_received() auf.

Die prüft, ob das Paket eine Anfrage ist. In diesem Fall enthält das Feld op den Wert 1. Außerdem stellt es sicher, dass die MAC-Zieladresse 00:00:00:00:00:00 (Broadcast) ist und die MAC-Adresse des Absenders mit der des Dash Button übereinstimmt (siehe Kasten auf S. 176).

Das Skript links gibt dann „Button gedrückt!“ aus. An dieser Stelle können Sie ansetzen, um beliebige andere Aktionen auszulösen. Zum Beispiel könnte das Skript mithilfe des Python-Moduls smtplib eine Mail verschicken oder mit requests beliebige URLs aufrufen oder Aktionen via IFTTT triggern [1].

Die nötige Software installieren Sie unter Raspbian mit folgenden Befehlen:

```
sudo apt install python-scapy tcpdump
sudo apt install python-requests
```

Das vollständige Skript finden Sie in unserem Git-Repository (siehe c't-Link). Es enthält Code für beide Module und prüft zusätzlich, ob der Button innerhalb der letzten fünf Sekunden gedrückt wurde. Das haben wir eingebaut, weil wir festgestellt haben, dass der Dash Button in seltenen Fällen mehrere ARP-Pakete nach dem Booten verschickt. Die Überprüfung verhindert so doppelte Reaktionen auf einen Knopfdruck.

Das Modul scapy braucht Administratorrechte. Deshalb müssen Sie das Skript mit sudo python2 listen.py als Root starten. Scapy gibt es nur für Python 2, aber nicht für Python 3. Mit dem Python 2.7 aus der Raspbian-Distribution läuft es klaglos.

## Grätsche rechts

Im bisherigen Fall verbindet sich der Button mit dem heimischen WLAN. Die Pakete des Buttons erreichen dann aber auch die Server von Amazon. Das ist kein Drama, denn solange Sie bei der Einrichtung des Buttons kein konkretes Produkt ausgewählt haben, führt Amazon keine Bestellung aus. Trotzdem bekommt Amazon mit, dass da ein Knopf gedrückt wurde. Außerdem nervt Sie die Amazon-App mit Benachrichtigungen, dass ein Bestellvorgang fehlgeschlagen ist.

Um das zu unterbinden, könnten Sie die MAC-Adressen-Filter auf Ihrem

Router konfigurieren, damit der die Pakete des Buttons nicht ins Internet weiterleitet. In der Router-Konfiguration finden Sie die Einstellungen üblicherweise unter „Kindersicherung“. Das kann aber schnell unübersichtlich werden, wenn man viele Dash Buttons zweckentfremden will.

Besser ist es, mithilfe des Raspi ein separates WLAN ausschließlich für die Dash Buttons aufzuspannen. Damit sparen Sie sich die Einrichtung einer Firewall-Regel für jeden neuen Button. Leitet der Raspi keinerlei Pakete aus diesem WLAN ins Internet weiter, bekommt Amazon vom Button-Drücken nichts mehr mit.

Da der Raspi über Ethernet noch am Internet hängt, kann das Skript trotzdem Webhooks triggern oder Mails verschicken.

Damit der Raspi das WLAN für die Buttons aufspannen kann, müssen Sie ihn als WLAN-Access-Point und DHCP-Server einrichten. Die dafür erforderliche Software steckt in den Paketen hostapd

(WLAN AP) und dnsmasq (DNS- und DHCP-Server):

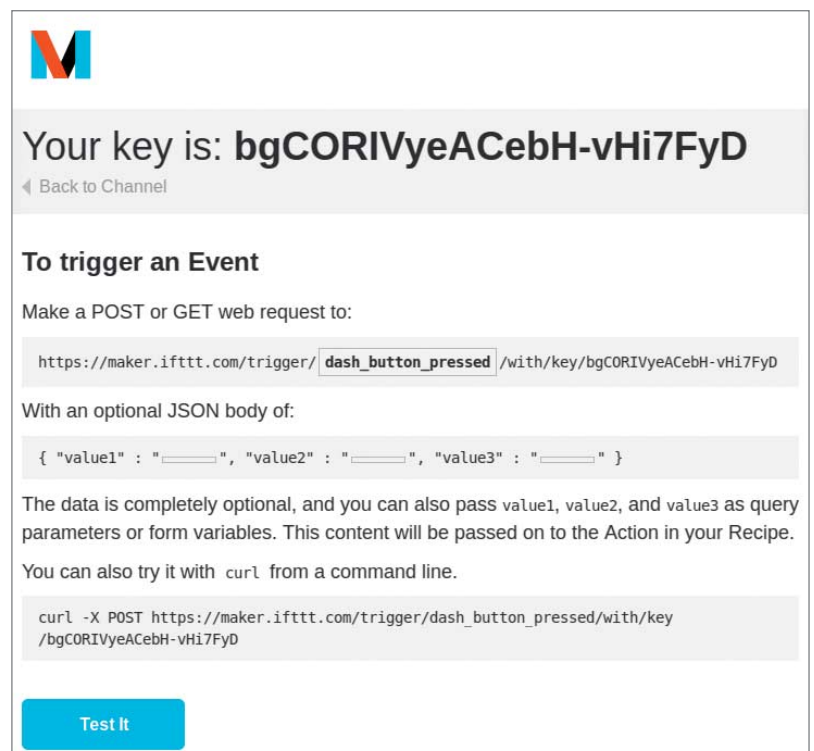
```
sudo apt install hostapd dnsmasq
```

Stellen Sie die WLAN-Schnittstelle anschließend in /etc/network/interfaces auf manuelle Konfiguration um:

```
allow-hotplug wlan0
iface wlan0 inet static
    address 192.168.12.1
    netmask 255.255.255.0
    network 192.168.12.0
    broadcast 192.168.12.255
```

Diese Zeilen ersetzen den Block mit iface wlan0 in der Konfigurationsdatei. Sie sorgen dafür, dass der NetworkManager die Finger vom WLAN lässt. Er bietet unter Raspbian nämlich standardmäßig keine Möglichkeit, ein eigenes Netz aufzuspannen.

Das aufzuspannende Netz konfigurieren Sie in /etc/hostapd/hostapd.conf. Legen Sie die Datei mit folgenden Einstellungen an:



**Your key is: bgCORIVyeACebH-vHi7FyD**

◀ Back to Channel

### To trigger an Event

Make a POST or GET web request to:

```
https://maker.ifttt.com/trigger/dash_button_pressed/with/key/bgCORIVyeACebH-vHi7FyD
```

With an optional JSON body of:

```
{ "value1" : " ", "value2" : " ", "value3" : " " }
```

The data is completely optional, and you can also pass value1, value2, and value3 as query parameters or form variables. This content will be passed on to the Action in your Recipe.

You can also try it with curl from a command line.

```
curl -X POST https://maker.ifttt.com/trigger/dash_button_pressed/with/key/bgCORIVyeACebH-vHi7FyD
```

**Test It**

Den Maker-Channel von IFTTT triggert ein Request an eine persönliche URL. Wenn Sie https://ifttt.com/maker/ aufrufen und dort auf „How to Trigger Events“ klicken, zeigt der Dienst Ihre persönliche URL an. In die fügen Sie nur noch den Namen Ihres Triggers ein und nutzen die URL im requests-Aufruf im Python-Skript.

## Button hack-fertig einrichten

Richten Sie Ihren Dash-Button wie von Amazon beschrieben ein. Das geht nur mit der Smartphone-Version der Amazon-Shopping-App. Die Tablet-Version bietet bisher keinen Menüpunkt für Dash-Buttons an. Die Schritt-für-Schritt-Anleitung fordert Sie dann auf, den Button sechs Sekunden lang zu drücken, bis die LED blau blinkt. In der App tragen Sie anschließend die SSID und das Passwort Ihres WLANs ein. Die App überträgt diese Konfiguration auf den Button, der sich daraufhin abschaltet. In

der App gelangen Sie dann zu einer Auswahlliste verschiedener Packungsgrößen zu dem auf dem Button aufgedruckten Produkt. Wenn Sie die App jetzt schließen und kein Produkt auswählen, bestellt der Button nichts, wenn Sie ihn drücken.

Sie kommen auch an diesen Punkt, wenn Sie schon mal ein Produkt ausgewählt und bestellt hatten. Löschen Sie den Button dafür aus der App und richten ihn neu ein. Ob die Produktauswahl fehlt, erkennen Sie auch am Button:

Wenn die LED nach dem Druck erst weiß wird und anschließend in ein wildes rotes Blinken übergeht, konnte Amazon keine Bestellung ausführen. Leuchtet die LED am Ende freundlich grün, konnte Amazon die Bestellung registrieren. Wenn Sie das nicht wollen, sollten Sie direkt stornieren.

Damit das Skript den Button erkennt, müssen Sie noch dessen MAC-Adresse herausfinden. Rufen Sie es dafür einmal ohne Änderungen auf und drücken Sie den Button. Das Skript gibt dann die MAC-Adressen von zwei unbekannten Geräten aus: Zuerst die des Buttons und danach die Ihres Raspi, der den Button sucht, nachdem der sich wieder abgeschaltet hat. Beenden Sie das Skript mit Strg+C und übertragen Sie die angezeigten MAC-Adressen in dessen Python-Code. Tauchen bei Ihnen noch mehr MAC-Adressen auf, liegt das daran, dass sich weitere Geräte wie Handys im gleichen Netz anmelden. Das Skript kann die Geräte nicht unterscheiden und zeigt auch deren MAC-Adressen. Die richtige Adresse erscheint ungefähr 4 Sekunden nach jedem Drücken.

```
pi@raspberrypi: ~/dashbutton
File Edit Tabs Help
pi@raspberrypi:~/dashbutton $ sudo python2 listen.py
Listening for ARP packets...
Unknown Device connecting: 50:f5:da:6f:98:6c
Unknown Device connecting: 50:f5:da:6f:98:6c
Unknown Device connecting: b8:27:eb:17:d5:22
Unknown Device connecting: b8:27:eb:17:d5:22
Unknown Device connecting: b8:27:eb:17:d5:22
Unknown Device connecting: b8:27:eb:17:d5:22
Unknown Device connecting: b8:27:eb:17:d5:22
```

Rufen Sie das Skript ohne Anpassung auf, zeigt es beim Drücken zuerst die MAC-Adresse des Buttons und danach die des Raspi. Beides übertragen Sie einfach per Copy&Paste in den Python-Code.

```
interface=wlan0
driver=nl80211
ssid=raspi-iot
hw_mode=g
channel=8
wpa=2
wpa_passphrase=BitteEinGutesPasswort!
wpa_key_mgmt=WPA-PSK
wpa_pairwise=CCMP
rsn_pairwise=CCMP
```

Falls Sie einen älteren Raspi mit USB-WLAN-Interface benutzen, müssen Sie die Angabe `driver=nl80211` auf den zu Ihrem WLAN-Chip passenden Wert ändern. Auf dem Raspi 3 stimmt die Einstellung bereits.

Um `hostapd` beim Boot zu starten, tragen Sie den Pfad zur Konfigurationsdatei in `/etc/default/hostapd` ein:

```
DAEMON_CONF=/etc/hostapd/hostapd.conf
```

Mit diesen Einstellungen erzeugt der Raspi ein WLAN, vergibt aber Geräten in diesem Funknetz keine IP-Adressen. Darum kümmert sich `dnsmasq`, dessen Konfiguration `/etc/dnsmasq.conf` Sie auch anpassen müssen. Mit folgendem Eintrag vergibt der Raspi die IP-Adressen im angegebenen Bereich für 12 Stunden (12h):

```
interface=wlan0
dhcp-range=192.168.12.50,192.168.12.150,12h
```

Die anderen Einstellungen belassen Sie einfach auf den voreingestellten Werten.

Ohne Internetverbindung schließt die Amazon-App die Einrichtung des Buttons leider nicht ab. Um den Raspi dazu zu

bringen, die Pakete aus dem WLAN an die Amazon-Server weiterzuleiten, aktivieren Sie IP-Forwarding mit folgenden Zeilen in `/etc/sysctl.conf`:

```
net.ipv4.ip_forward=1
net.ipv6.conf.all.forwarding=1
```

Nach einem Neustart betreibt Ihr Raspi sein eigenes WLAN, leitet aber weiterhin noch keine Pakete daraus ins Internet weiter. Dafür müssen Sie nämlich zusätzlich eine temporäre Firewall-Regel mit folgendem Befehl setzen:

```
sudo iptables -t nat -A POSTROUTING \
-o eth0 -j MASQUERADE
```

Konfigurieren Sie danach Ihren Dash-Button so, dass er das WLAN vom Raspi verwendet. Dank IP-Forwarding klappt

die Einrichtung, Amazon erfährt davon aber auch. Da der Raspi die Firewall-Regel nicht dauerhaft speichert, unterbinden Sie weitere Kommunikation mit den Amazon-Servern aber einfach mit einem Neustart. Die Einträge in `/etc/sysctl.conf` müssen Sie dafür nicht wieder auskommentieren: Ohne die temporäre Firewall-Regel verirrt sich kein Paket aus dem WLAN ins Internet.

Die vom Dash Button verschickten Pakete stranden nach dem Neustart im abgeschotteten Funknetz des Raspi und Amazon erfährt von den Tastendrücken nichts. Falls Sie weitere Buttons hinzufügen möchten, führen Sie einfach den `iptables`-Befehl aus und richten die Buttons ein. Ein Neustart trennt alle Buttons danach wieder vom Internet.

## Ausblick

Außer für das eingangs geschilderte Guten-Morgen-Szenario kann man den Button noch für viele andere Zwecke einsetzen: Besitzer von Philips-Hue-Lampen sollten einen Blick auf die Python-Bibliothek `phue` werfen, die die Lampen mit wenig Aufwand ansteuert. Aber Obacht: Vom Drücken des Knopfs bis zur Einschalten der Lampe vergehen wegen des Boot-Vorganges zirka 4 Sekunden. Die Verzögerung ergibt sich einzig aus dem Starten der Dash-Button-Firmware.

In den Weiten des Internets sind viele weitere Ideen aufgetaucht, was man mit den billigen Knöpfen alles machen kann. Ein Musiker drückt ihn beispielweise immer bevor und nachdem er sein Instrument übt. Die Statistik schreibt er automatisiert in ein Google-Spreadsheet zur späteren Auswertung. Ein junger Vater drückt einen Dash-Button immer, wenn er nachts vom schreienden Baby geweckt wird. Die Statistik soll ihm dabei helfen, den für das Kind perfekten Tagesrhythmus zu finden. Wir liebäugeln damit, einen Knopf neben die Kaffee-

maschine in der Redaktionsküche zu kleben, der den Techniker informiert, dass die Maschine ausgefallen ist. ([jme@ct.de](mailto:jme@ct.de)) **ct**

## Literatur

[1] Jo Bager, Der Alles-Automatisierer, Routineaufgaben im Netz automatisieren mit IFTTT, c't 5/15, S. 130

Das Skript bei GitHub: [ct.de/ykr3](https://github.com/ct.de/ykr3)

Anzeige