Hi, guys,

since I have my dyndns entries running at Cloudflare, but Cloudflare is not included in the list of providers, I simply extended the file /usr/lib/python2.7/site-packages/ddns/providers.py.

I based it on the script by Adrien Brignon (https://github.com/adrienbrignon/cloudflare-ddns 37)

Unfortunately, I don't know much about Python at all and have managed to create something that works with a lot of trial and error.
Unfortunately I had to install the python library requests after all.
(https://forum.ipfire.org/viewtopic.php?t=14349#p87989 17)

```
pakfire install python-distutils
wget https://bootstrap.pypa.io/get-pip.py
python ./get-pip.py
pip install requests
```

Only with the library urllib2 (i.e. with on-board tools) I haven't got it right yet.

If anyone can use it, here is the code snippet to build it in yourself:
Don't forget to include library requests!

```
#!/usr/bin/python
###############################################################################
###
#
#
# ddns - A dynamic DNS client for IPFire
#
# Copyright (C) 2012-2017 IPFire development team
#
...
#
import requests
...
...
    class DDNSProviderCloudflare(DDNSProvider):
    handle    = "cloudflare.com"
    name      = "Cloudflare"
    website   = "https://www.cloudflare.com"
    protocols = ("ipv4",)

    # Detailed information about the update request can be found here:
    # https://api.cloudflare.com/#dns-records-for-a-zone-update-dns-record

    # Details about the possible response codes:
    # https://api.cloudflare.com/#dns-records-for-a-zone-export-dns-records

    url = "https://api.cloudflare.com/client/v4/"
    can_remove_records = False
```

```
    def update_protocol(self, proto):
        API_HEADERS = {
            'X-Auth-KEY': self.password,
            'X-Auth-Email': self.username
        }

        host, domain = self.hostname.split(".", 1)

        payload = {
            'name': domain
        }
        r = requests.get(self.url + "zones", headers=API_HEADERS,
params=payload)
        data = r.json().get("result")
        zone_id = data[0]["id"]
        zone_name = data[0]["name"]

        payload = {
            "page": 0,
            "per_page": 50
        }
        r = requests.get(self.url + "zones/" + zone_id + "/dns_records/",
headers=API_HEADERS, params=payload)
        data = r.json().get("result")
        for i in range(len(data)):
            if self.hostname in data[i]["name"] and data[i]["type"] == "A":
                host_id = data[i]["id"]
                payload = {
                    "type": data[i]["type"],
                    "name": self.hostname,
                    "content": self.get_address(proto),
                    "ttl": data[i]["ttl"],
                    "proxied": data[i]["proxied"]
                }
            r = requests.put(self.url + "zones/" + zone_id + "/dns_records/"
+ host_id, headers=API_HEADERS, json=payload)
            success = r.json().get("success")
            # Handle success messages.
            if success:
                return
            else:
                raise DDNSError
        if self.hostname != data[i]["name"]:
            raise DDNSRequestError(_("Invalid hostname specified"))
```