Android VPN

- Am Beispiel eines Netzwerktunnels für das Domain Name System (DNS)

Inhalt

- VPN Framework in Android
- Übersicht zu DNS
- Iodine Funktionsweise
- Demonstration

VPN und Android

- Verfügbar seit Android 4.0
- API Documentation:
  http://developer.android.com/reference/android/net/VpnService.html
- Stellt einen Filedescriptor wie in Linux TUN Devices zur Verfügung
- Implementierung von VPN Clients in Java oder auch mittels JNI in C möglich

VPN und Android

```
IodineVpnService::runTunnel()
    Builder b = new VpnService.Builder();
    b.addAddress(hostAddress, netbits);
    b.addRoute("0.0.0.0", 0); // Default Route
    b.setMtu(mtu);

    // Opens tun device
    ParcelFileDescriptor parcelFD = b.establish();

    // prevent dns traffic to get through its own tunnel
    protect(IodineClient.getDnsFd());

    // get the filedescriptor
    int tun_fd = parcelFD.detachFd();

    // pass the filedescriptor to iodine
    IodineClient.tunnel(tun_fd);
```

```
public class IodineClient {
    public static native tunnel(int fd);
    …
}
```

VPN und Android

jni/iodine-client.c

```
…
JNIEXPORT jint JNICALL
Java_org_xapek_andiodine_IodineClient_tunnel(JNIEnv *env,
        jclass klass, jint tun_fd) {

    printf("Run client_tunnel_cb");
    int retval = client_tunnel_cb(tun_fd, dns_fd, &tunnel_continue_cb);

    close(dns_fd);
    close(tun_fd);
    return retval;
}
…
```

DNS – typisches Szenario

Web Browser
http://example.com

A www.example.com

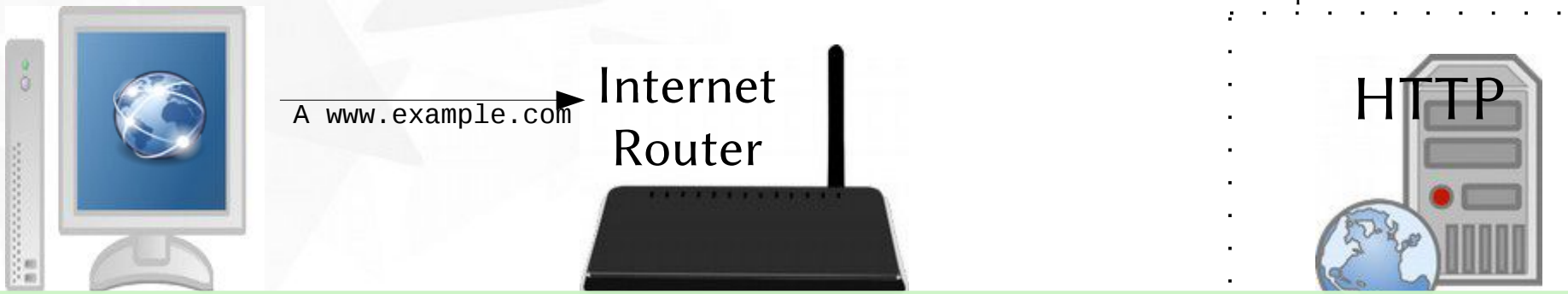
Internet
Router

Provider DNS

example.com

.com TLD: Verisign

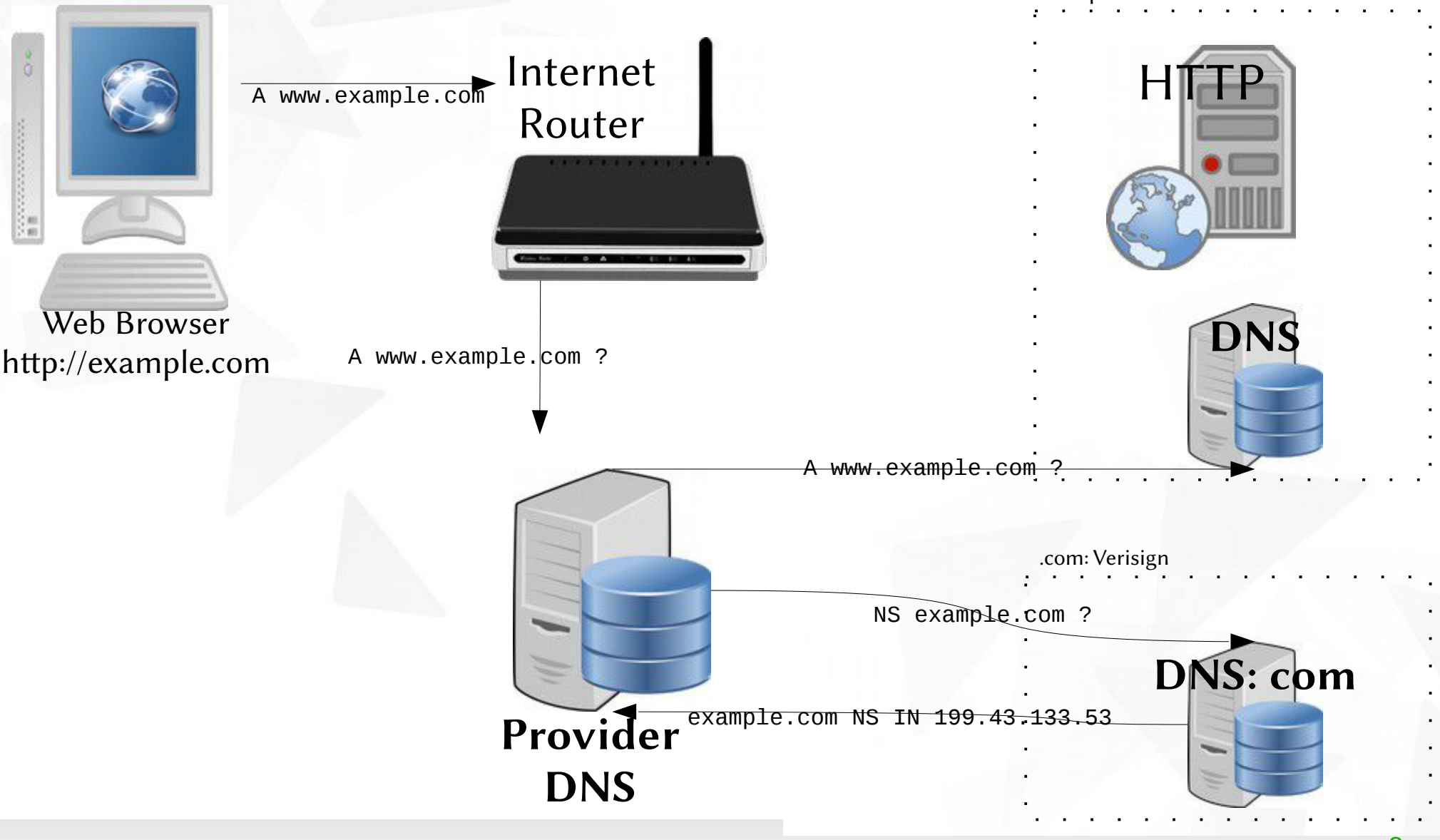
DNS – typisches Szenario



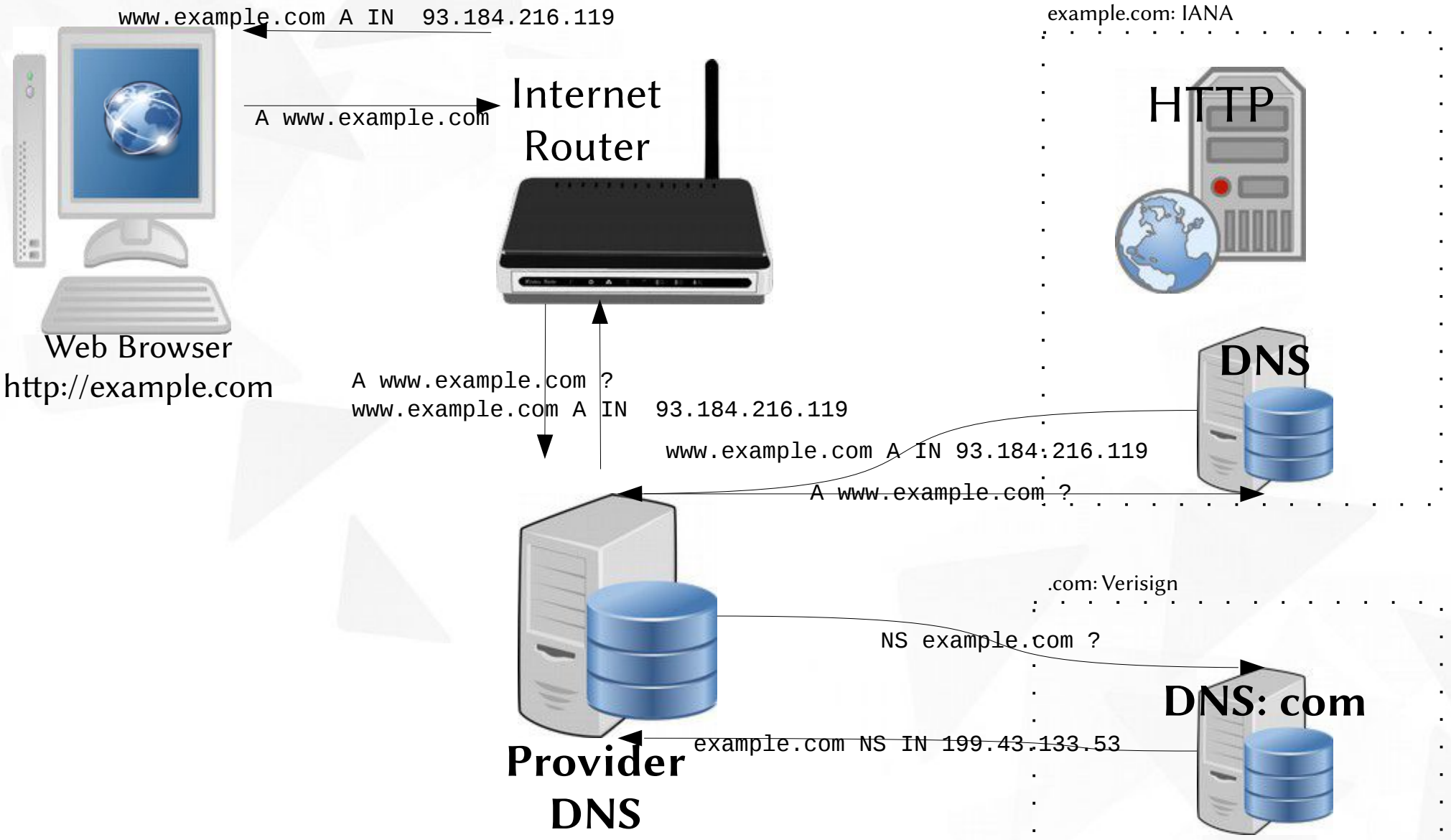

```
Time         Source      Destination  Protocol Info
2.425301000  10.1.1.145  10.1.0.1     DNS      Standard query 0x5e9e A example.com

Internet Protocol Version 4, Src: 10.1.1.145 (10.1.1.145), Dst: 10.1.0.1 (10.1.0.1)
User Datagram Protocol, Src Port: 52963 (52963), Dst Port: domain (53)
Domain Name System (query)
    [Response In: 16]
    Transaction ID: 0x5e9e
    Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
    Queries
        example.com: type A, class IN
            Name: example.com
            Type: A (Host address)
            Class: IN (0x0001)
```

DNS – typisches Szenario



DNS – typisches Szenario

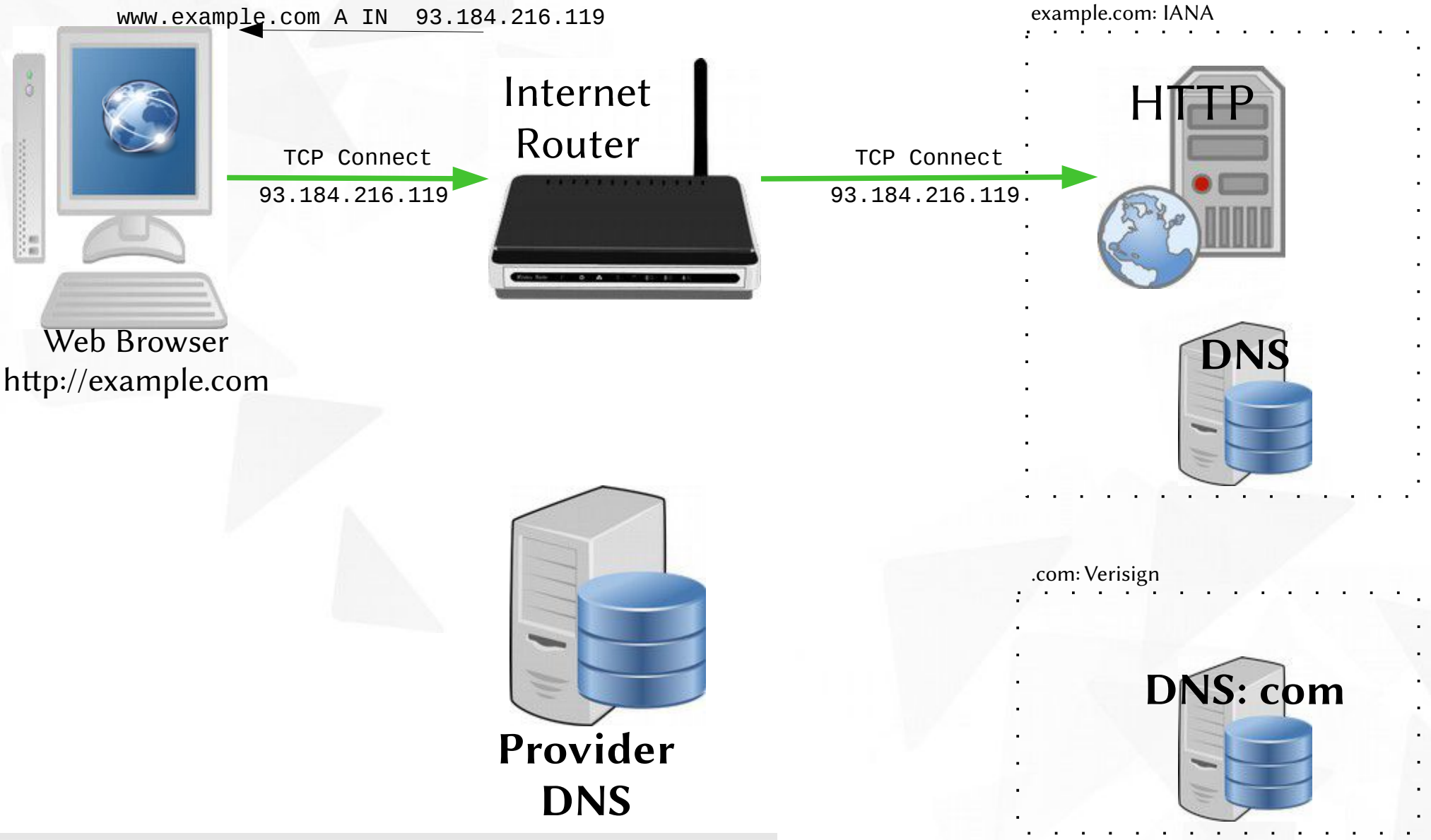


DNS – typisches Szenario

```
Time         Source    Destination  Protocol Info
2.493068000  10.1.0.1  10.1.1.145   DNS      query response 0x5e9e A 93.184.216.119

Internet Protocol Version 4, Src: 10.1.0.1 (10.1.0.1), Dst: 10.1.1.145 (10.1.1.145)
User Datagram Protocol, Src Port: domain (53), Dst Port: 52963 (52963)
Domain Name System (response)
    Transaction ID: 0x5e9e
    Flags: 0x8180 Standard query response, No error
    Questions: 1
    Answer RRs: 1
    Authority RRs: 0
    Additional RRs: 0
    Queries
        bla.de: type A, class IN
            Name: example.com
            Type: A (Host address)
            Class: IN (0x0001)
    Answers
        example.com: type A, class IN, addr 93.184.216.119
            Name: example.com
            Type: A (Host address)
            Class: IN (0x0001)
            Time to live: 5 hours, 39 minutes, 47 seconds
            Data length: 4
            Addr: 93.184.216.119  (93.184.216.119)
```

DNS – typisches Szenario



Nutzdaten über DNS

- DNS ist nicht für beliebige Nutzdaten ausgelegt. Domainnamen:
  - Keine Groß/Kleinschreibung
  - Keine Sonderzeichen

- Kodierung der Nutzdaten erforderlich: Base32 (RFC 3548)
  - Kodierung beliebiger binärdaten in 32 Zeichen
  - Länge der Darstellung wächst um ca. 160%
  - Beispiel:

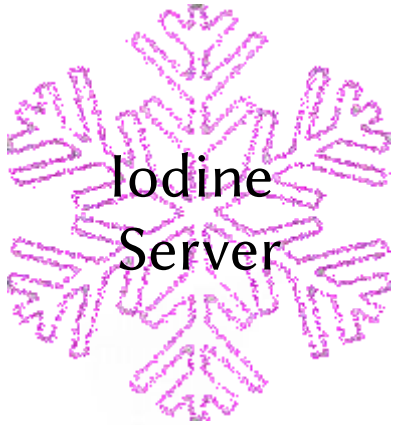
```
abcdefg (7 bytes)
Mfrggzdfmztq (12 bytes)
```

DNS – iodine Szenario

Endgerät mit
Iodine Client

Internet
Router

example.com




```
lad24srn4ezmg21qjsfy13msagd0srfq.t.example.com
```

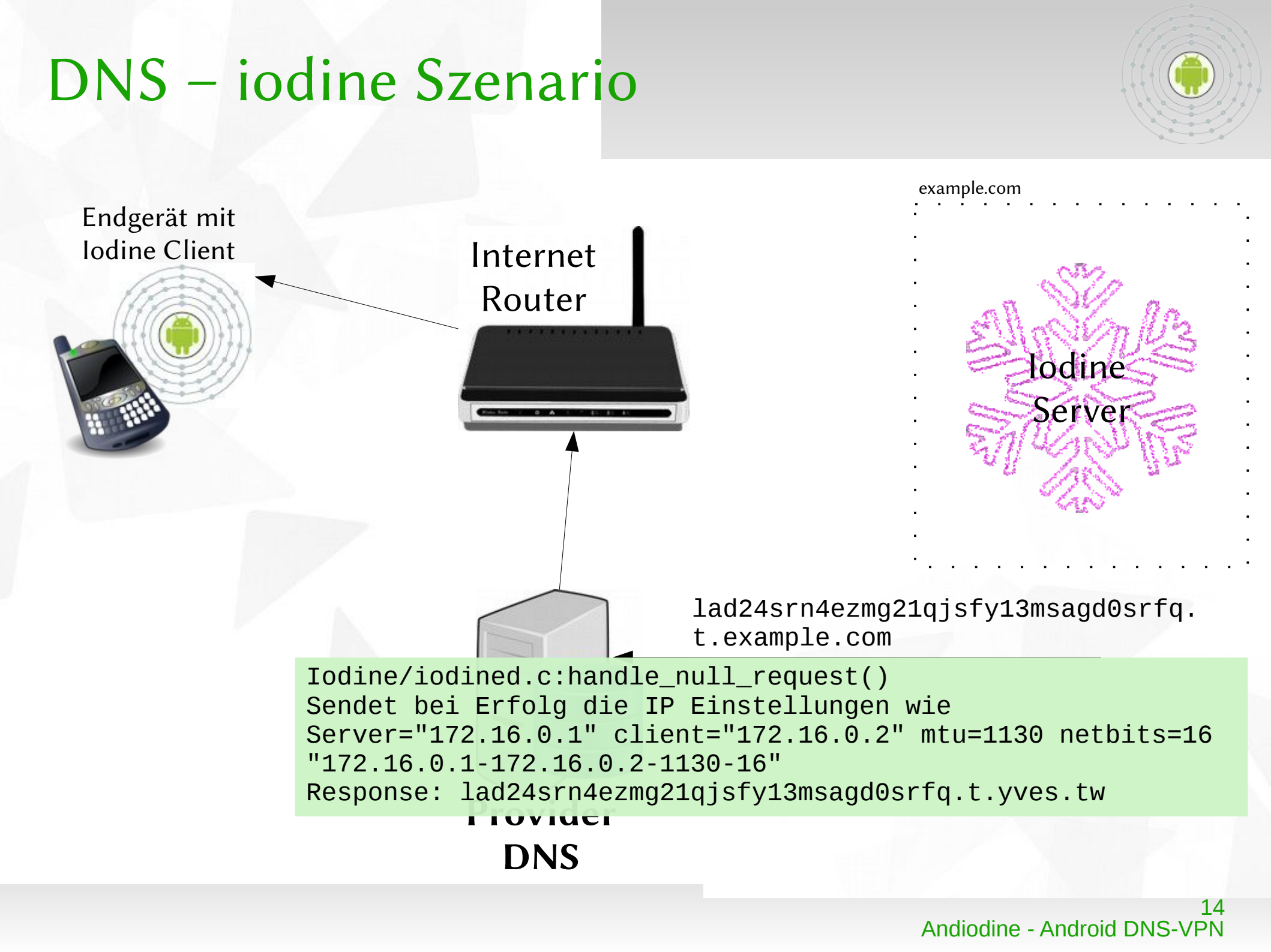
Iodine/client.c:send_login()

- Command 'l': Login
- hostname[1..16] = password mit seed xored und md5
- hostname[17..18] = seed

```
Request: lad24srn4ezmg21qjsfy13msagd0srfq.t.yves.tw
```

Provider DNS

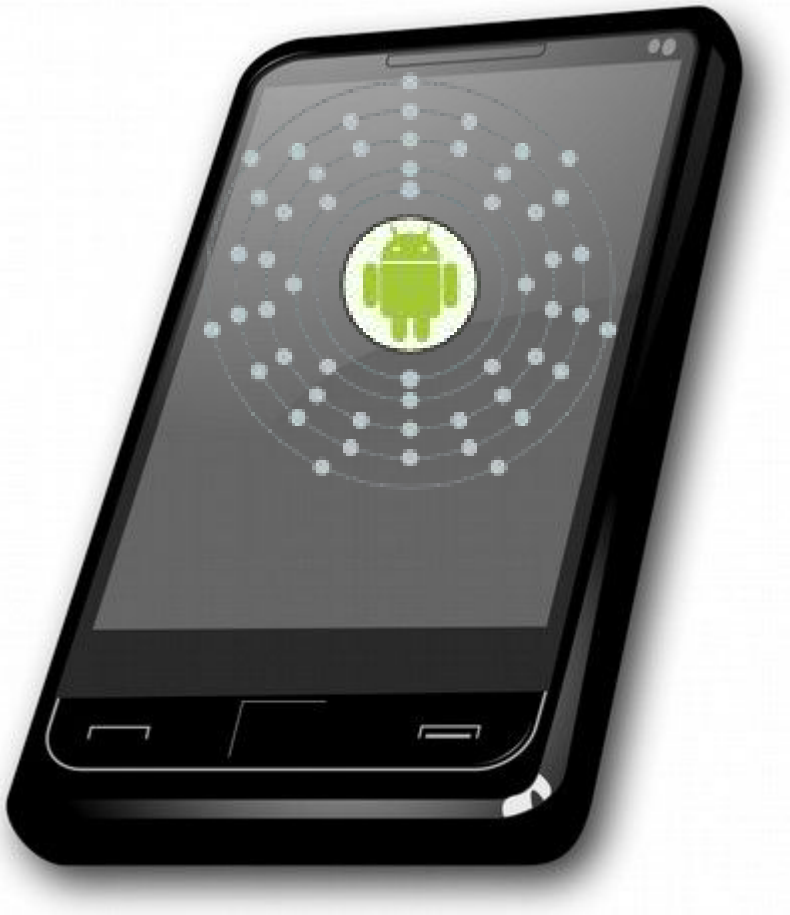
DNS – iodine Szenario



DNS – Tunnel Gegenmaßnahmen

- Große/Lange Requests ausbremsen
- Request Rate pro Zeiteinheit begrenzen (aber kurze Spitzen zulassen)
- Keine externen DNS Server erlauben, Port 53 UDP sperren
- Einzelne Anwendungen wie Iodine können anhand fester Muster leicht erkannt werden

Demonstration



Vielen Dank ...