FH MÜNSTER
University of Applied Sciences

ETI
FB Elektrotechnik und Informatik
Department of Electrical Engineering
and Computer Science

Portscanning & Servicefingerprinting mit Javascript

Seminar Informatik im Wintersemester 2016 / 2017
Betreut durch: Prof. Dr. Sebastian Schinzel & M. Sc. Christian Dresen

Jan Hölscher
Fachinformatiker für Systemintegration

E-Mail: hoelscher.jan@fh-muenster.de
Twitter: @wers_er_denn

Philipp Schwarz
Fachinformatiker für Anwendungsentwicklung

E-Mail: philipp.schwarz@fh-muenster.de
Twitter: @hawki1962

Agenda

1. Allgemeine Informationen zu Portscanning & Servicefingerprinting
2. Javascript und die Same-Origin-Policy
3. Aktueller Scan mittels Javascript / WebRTC
4. Zusammenfassung

FH MÜNSTER
University of Applied Sciences

Allgemeine Informationen

zu Portscanning & Servicefingerprinting

Portscanning & Servicefingerprinting

Allgemeine Erläuterung

FH MÜNSTER
University of Applied Sciences

Portscanning

- Information über Erreichbarkeit eines Hosts und seinen Services
- Scan des öffentlichen Internets trivial
- Scan eines Netzwerk hinter NAT Boxen nicht direkt möglich

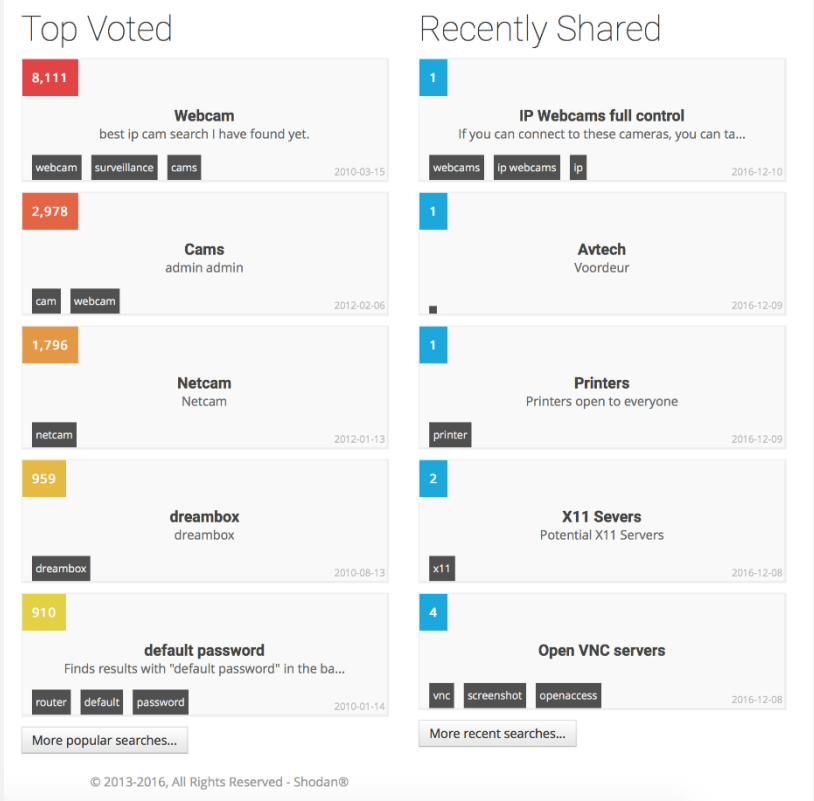
Servicefingerprinting

- Eindeutiges identifizieren einer Firmware
- Vorabanalysiere von Firmware
- Vgl. von Scan Daten mit generierten Fingerprints

Portscanning & Servicefingerprinting

Motivation der Durchführung

- Wird von verschiedenen Institutionen durchgeführt
  - FH-Münster
  - censys.io
  - shodan.io
- Analyse von verwendeten Geräten, z.B.
  - Router mit Fernwartungsschnittstelle TR-069
  - IP-Kameras mit Standard Passwörtern
- Entdecken von Sicherheitslücken
  - Forschung und Lehre: Informieren
  - Hacker und Angreifer: Lücken ausnutzen



https://www.shodan.io/explore

FH MÜNSTER
University of Applied Sciences

Javascript

und die Same-Origin-Policy

Javascript und Same-Origin-Policy

Allgemeine Erläuterung

- Clientseitige Skriptsprache
- In jedem aktuellen Browser

- Sicherheitskonzept für Browser und Webanwendungen
- Zugriff auf Objekte anderer Webseiten untersagt
- Eingeführt von Netscape 1996

Javascript
und SOP Einschränkungen

FH MÜNSTER
University of Applied Sciences

```
<p id="demo”></p>

<script>
  var xmlhttp = new XMLHttpRequest();
  xmlhttp.onreadystatechange = function() {
    document.getElementById("demo").innerHTML = this.status;
  };
  xmlhttp.open("GET", "http://www.example.org", true);
  xmlhttp.send();
</script>
```

Javascript
und SOP Einschränkungen

FH MÜNSTER
University of Applied Sciences

- Chrome Network Log:

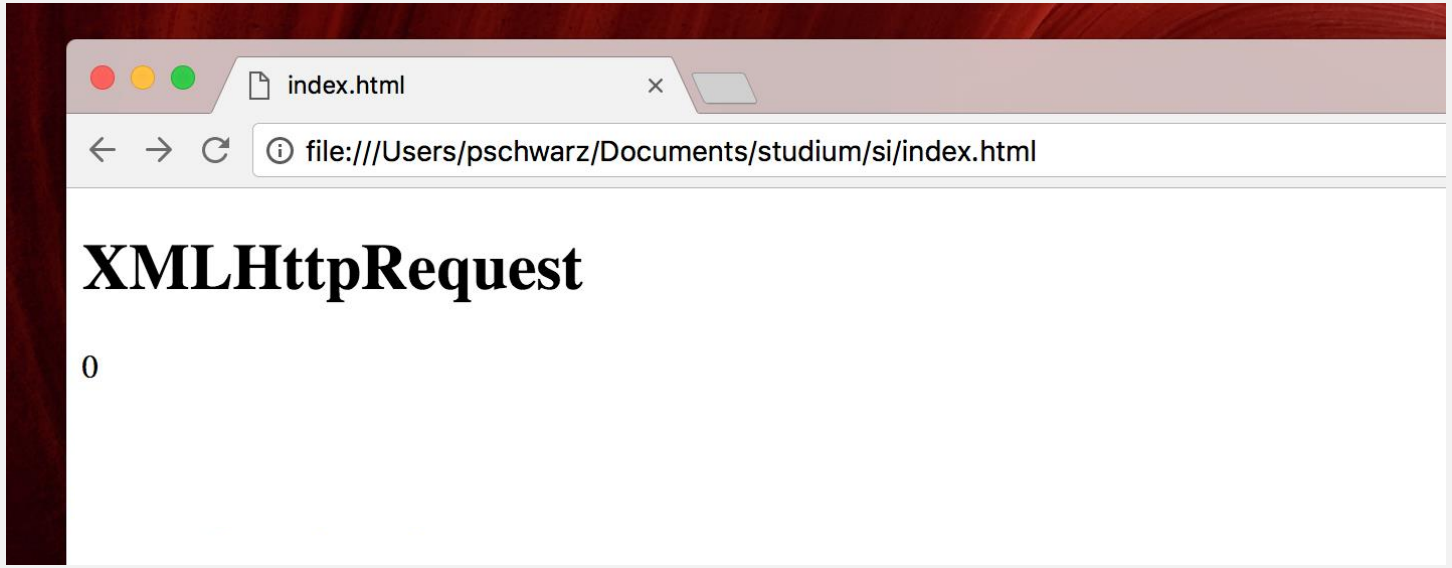
Name	Status	Type	Initiator	Size	Time
index.html	Finished	document	Other	0 B	5 ms
www.example.org	200	xhr	index.html:16	(from disk ...	2 ms

- Chrome Console Log:

  XMLHttpRequest cannot load http://www.example.org/.
  No 'Access-Control-Allow-Origin' header is present on the requested resource. Origin 'null' is therefore not allowed access.

Javascript
und SOP Einschränkungen

- “tatsächlicher” Wert von this.status

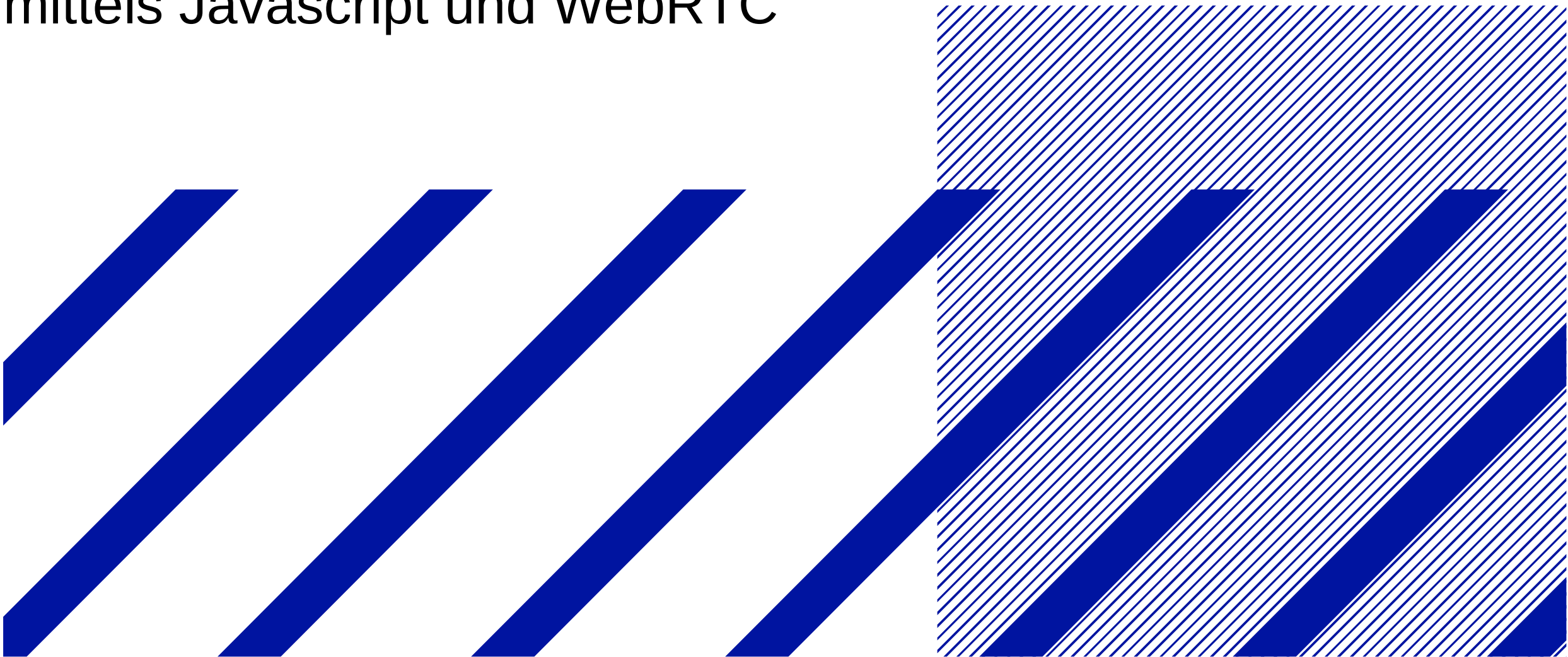


- getComputedStyle

FH MÜNSTER
University of Applied Sciences

Aktueller Scan

mittels Javascript und WebRTC



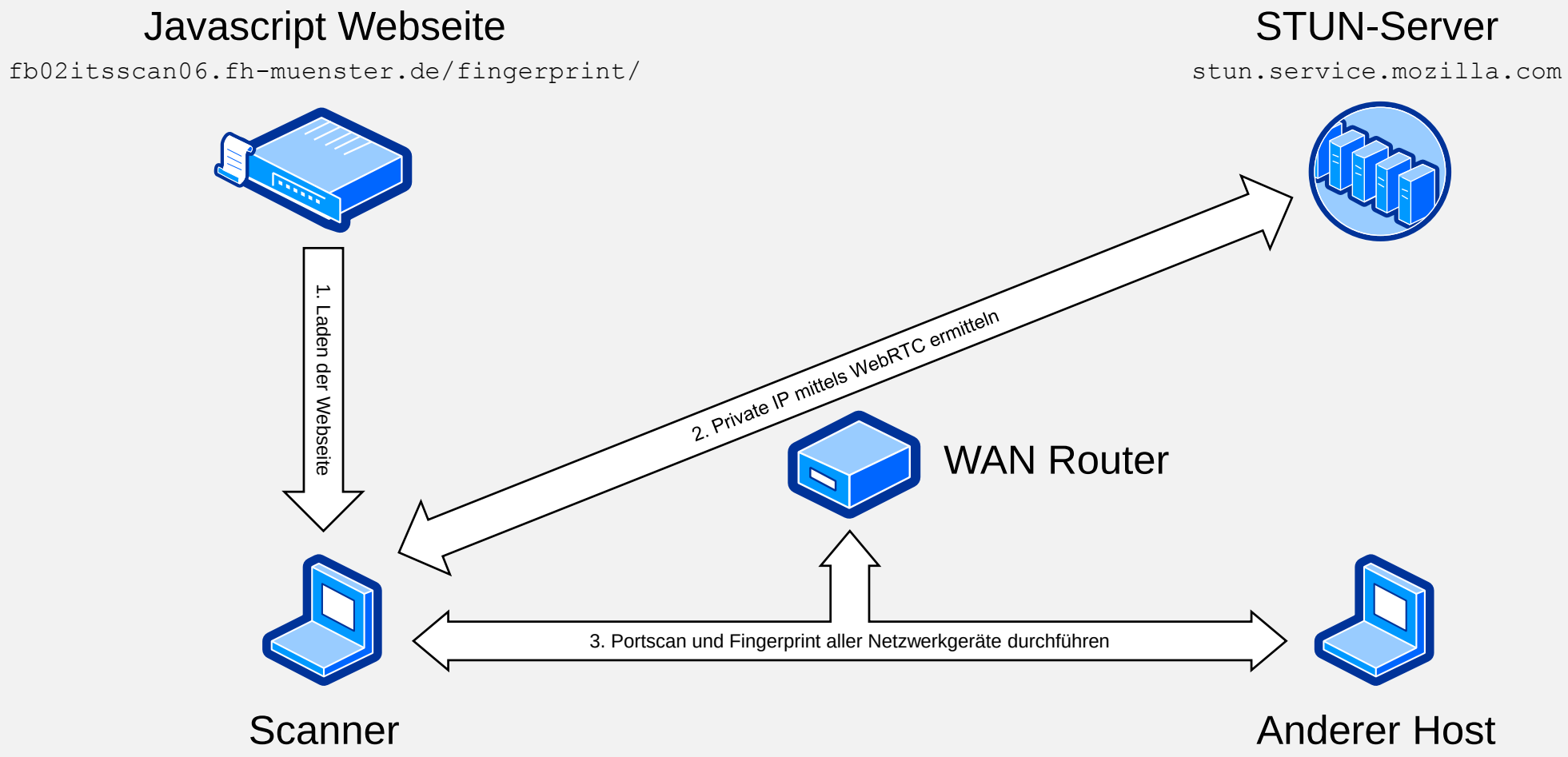
Aktueller Scan mittels Javascript & WebRTC

Technische Voraussetzungen

- Browser mit WebRTC Unterstützung
  - Microsoft Edge, Google Chrome (ab V.23), Mozilla Firefox (ab V.22), Opera (ab V.18)
- STUN Server
  - `stun.l.google.com`
  - `stun.services.mozilla.com`
- Webseite mit entsprechendem Javascript
  - `http://fb02itsscan06.fh-muenster.de/fingerprint/`

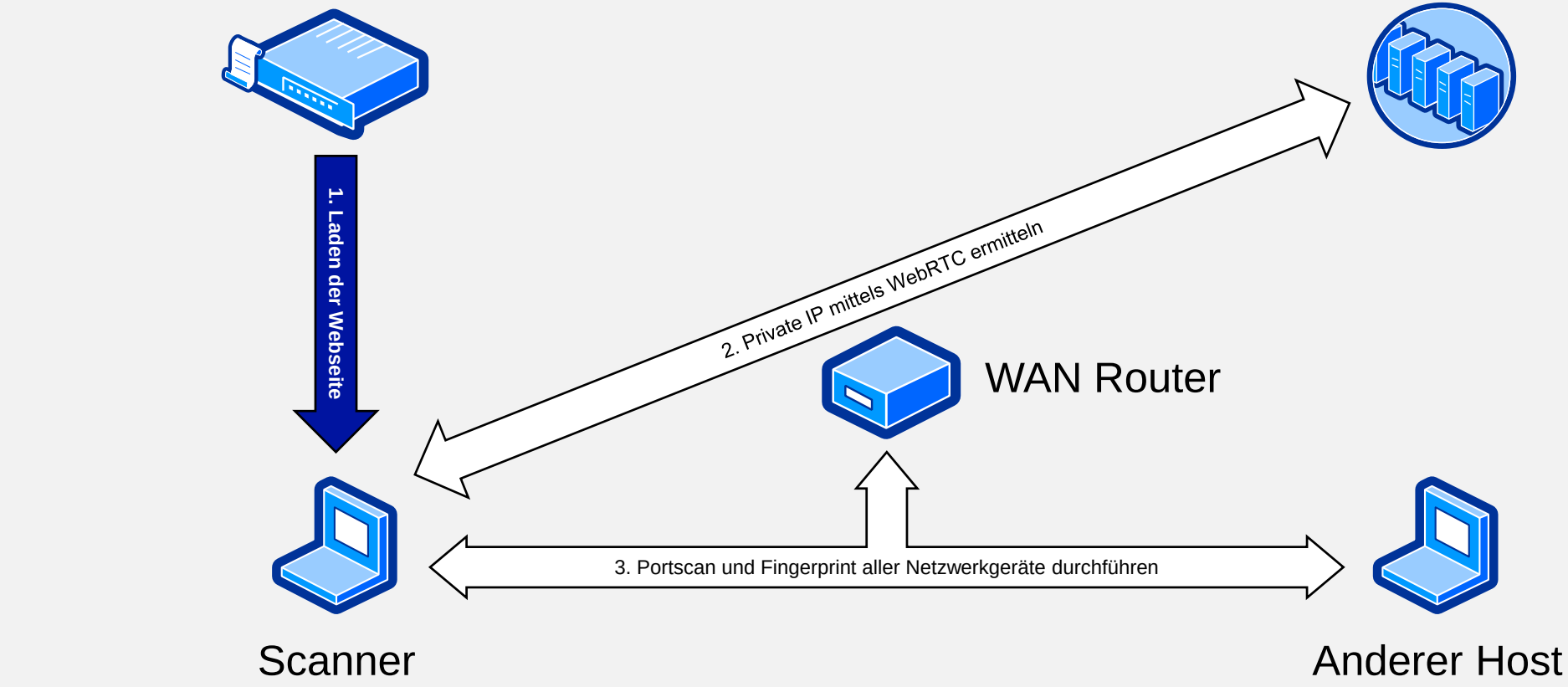
Aktueller Scan mittels Javascript & WebRTC

Ablauf des Scan



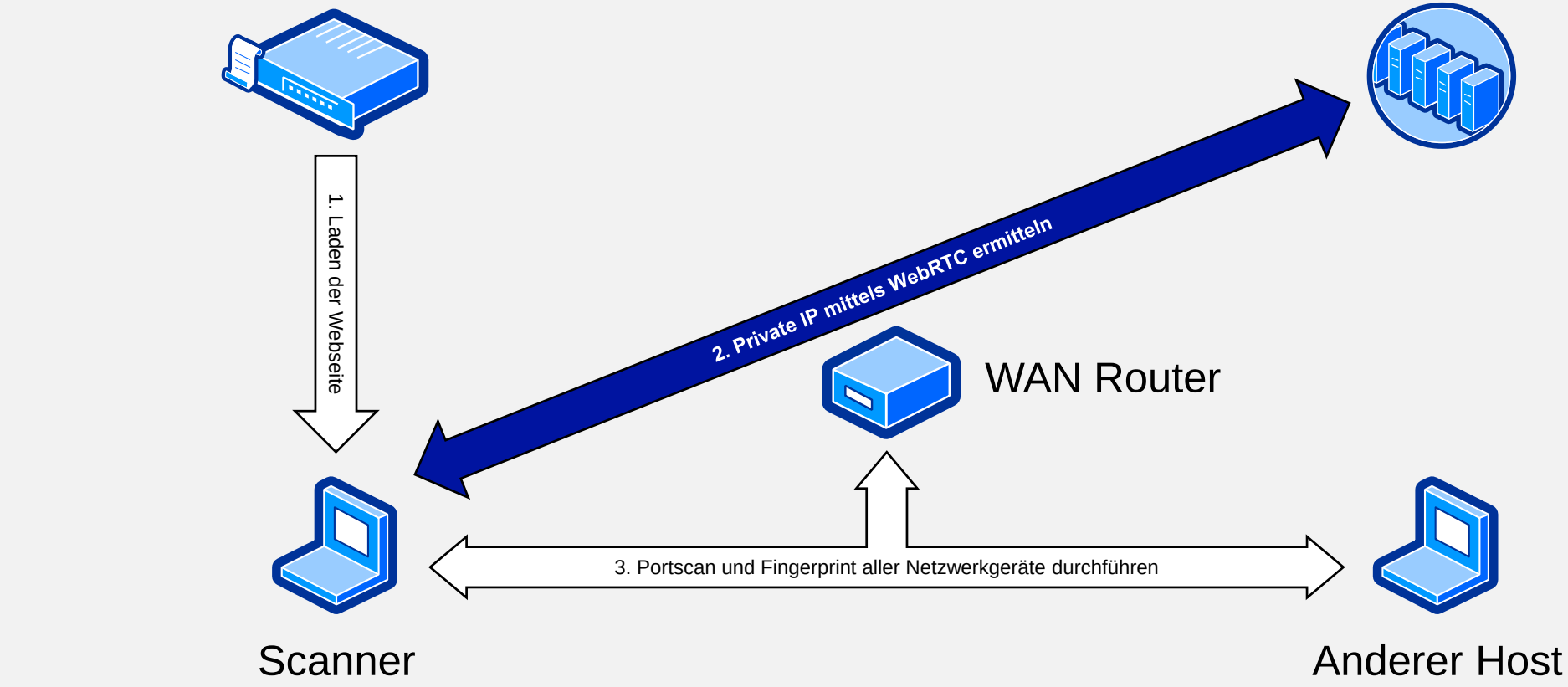
Aktueller Scan mittels Javascript & WebRTC

Ablauf des Scan



Aktueller Scan mittels Javascript & WebRTC

Ablauf des Scan

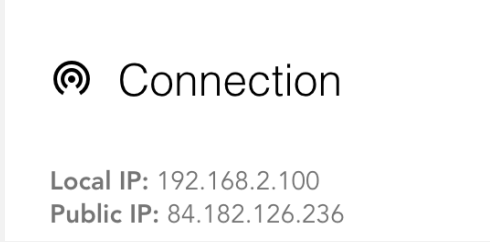


Aktueller Scan mittels Javascript & WebRTC

Technische Details – Javascript API WebRTC

- WebRTC als Standard des World Wide Web Consortium (W3C)
- Ermöglicht Peer-to-Peer Kommunikation für z.B. Videotelefonie

- Nutzt verschiedene Protokolle um NAT Boxen zu entdecken und zu „umgehen“
  - ICE: Interactive Connectivity Establishment
  - SDP: Session Description Protocol



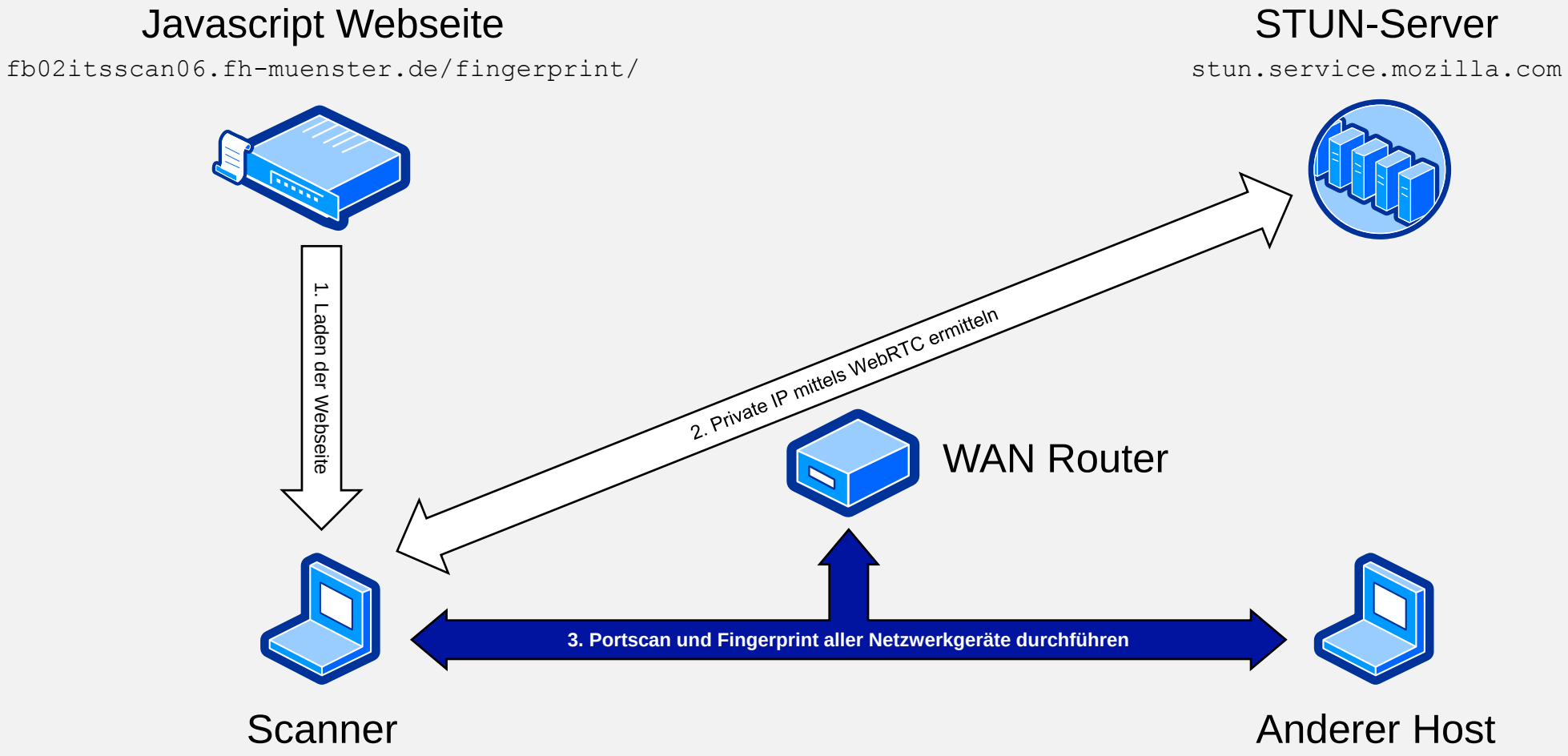
http://webkay.robinlinus.com

- Auszug aus einer SDP Nachricht:

```
a=candidate:1026183099 1 udp 2113937151 192.168.2.100  54461 typ host generation 0 network-cost 50
a=candidate:842163049  1 udp 1677729535 84.182.126.236 54461 typ srflx <gekürzt> generation 0 network-cost 50
```

Aktueller Scan mittels Javascript & WebRTC

Ablauf des Scan



Aktueller Scan mittels Javascript & WebRTC

Technische Details – Ping

- Script zum Pingen mit Javascript öffentlich verfügbar:
  - `https://github.com/jdfreder/pingjs`
- Laden einer zufälligen Resource mittels Javascript
  - z.B. `GET http://192.168.2.106/9584uc`
- Interpretieren verschiedener Return Parameter:
  - `<resource>.onload ,<resource>.onerror`
  - Kein Return Parameter → Timeout
- Unterscheidung, ob Host erreichbar oder nicht

Aktueller Scan mittels Javascript & WebRTC

Technische Details – Fingerprint

- Identifizieren von Firmware über bestimmte abrufbare Dateien
  - Optimal: Pro Firmware eine eindeutige Datei

- CSS und JS Dateien über `.onload` bzw. `.onerror`
- Bilder über `img.width > 0`

- Verknüpfen mit einer Datenbank mit gespeicherter Fingerprints

```
▶ GET http://192.168.2.106/image/logo_mts.gif net::ERR_CONNECTION_REFUSED 192.168.2.106/image/logo_mts.gif:1
▶ GET http://192.168.2.106/image/wifi_port_positive.png net::ERR_CONNECTION_REFUSED 192.168.2.106/image/wifi_port_positive.png:1
▶ GET http://192.168.2.106/css/default/images/mic.png net::ERR_CONNECTION_REFUSED 192.168.2.106/css/default/images/mic.png:1
▶ GET http://192.168.2.106/icon/Cable_STB_8.png net::ERR_CONNECTION_REFUSED 192.168.2.106/icon/Cable_STB_8.png:1
▶ GET http://192.168.2.106/web/pic/dlink_utility.jpg net::ERR_CONNECTION_REFUSED 192.168.2.106/web/pic/dlink_utility.jpg:1
▶ GET http://192.168.2.106/image/flags/lang_lit.png net::ERR_CONNECTION_REFUSED 192.168.2.106/image/flags/lang_lit.png:1
▶ GET http://192.168.2.106/settings_ca.bmp net::ERR_CONNECTION_REFUSED 192.168.2.106/settings_ca.bmp:1
```

Google Chrome 55, Console während Fingerprint von 192.168.2.106

Aktueller Scan mittels Javascript & WebRTC

Eigenschaften der Fingerprints

Bilder

- Pfad des Bildes
  - inkl. Format,
- Hashwert
- Größe in Pixeln

Javascript

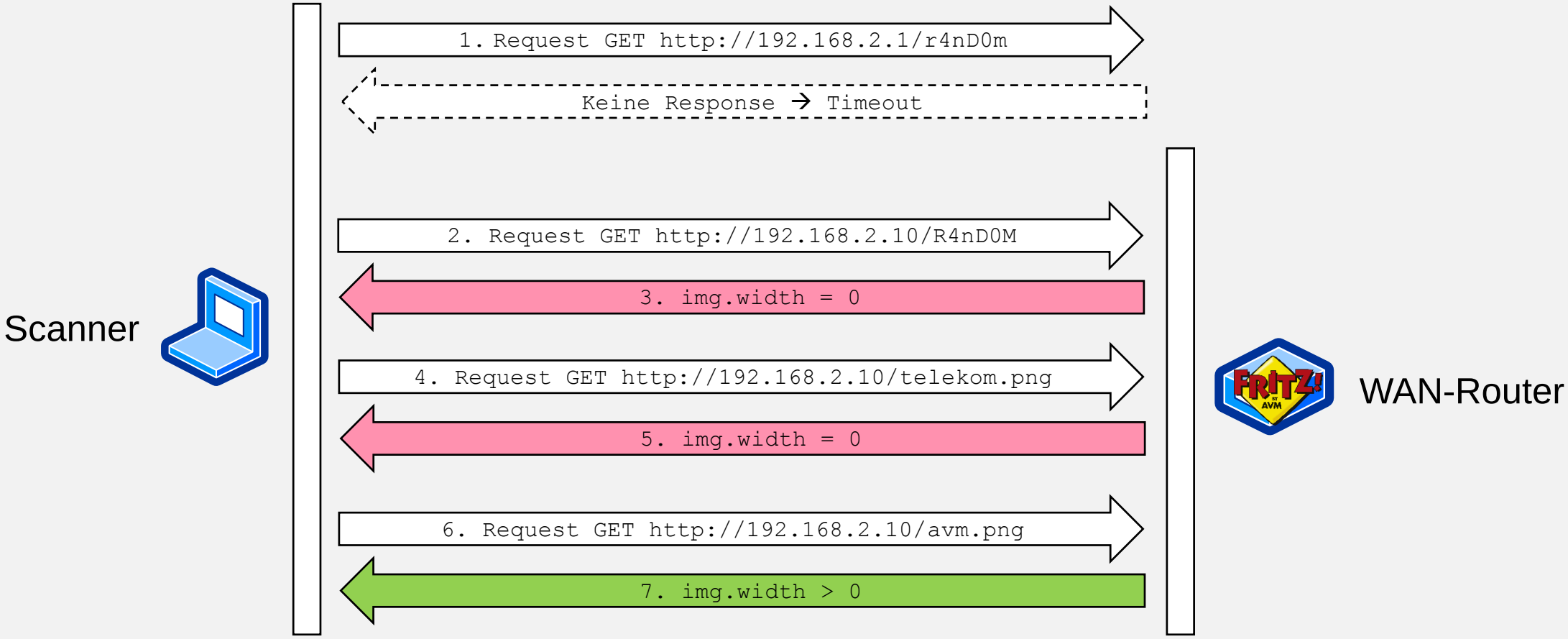
- Pfad der JS Datei
- Verfügbare Funktionen
- Verfügbare Variablen

CSS

- Pfad der CSS Datei
- Definierte Elementnamen
- Definierte Attribute

Aktueller Scan mittels Javascript & WebRTC

Minimal Beispiel eines Scans



Zusammenfassung

- Scan von lokalen Netzen möglich
- Angriffsszenario über Social Engineering

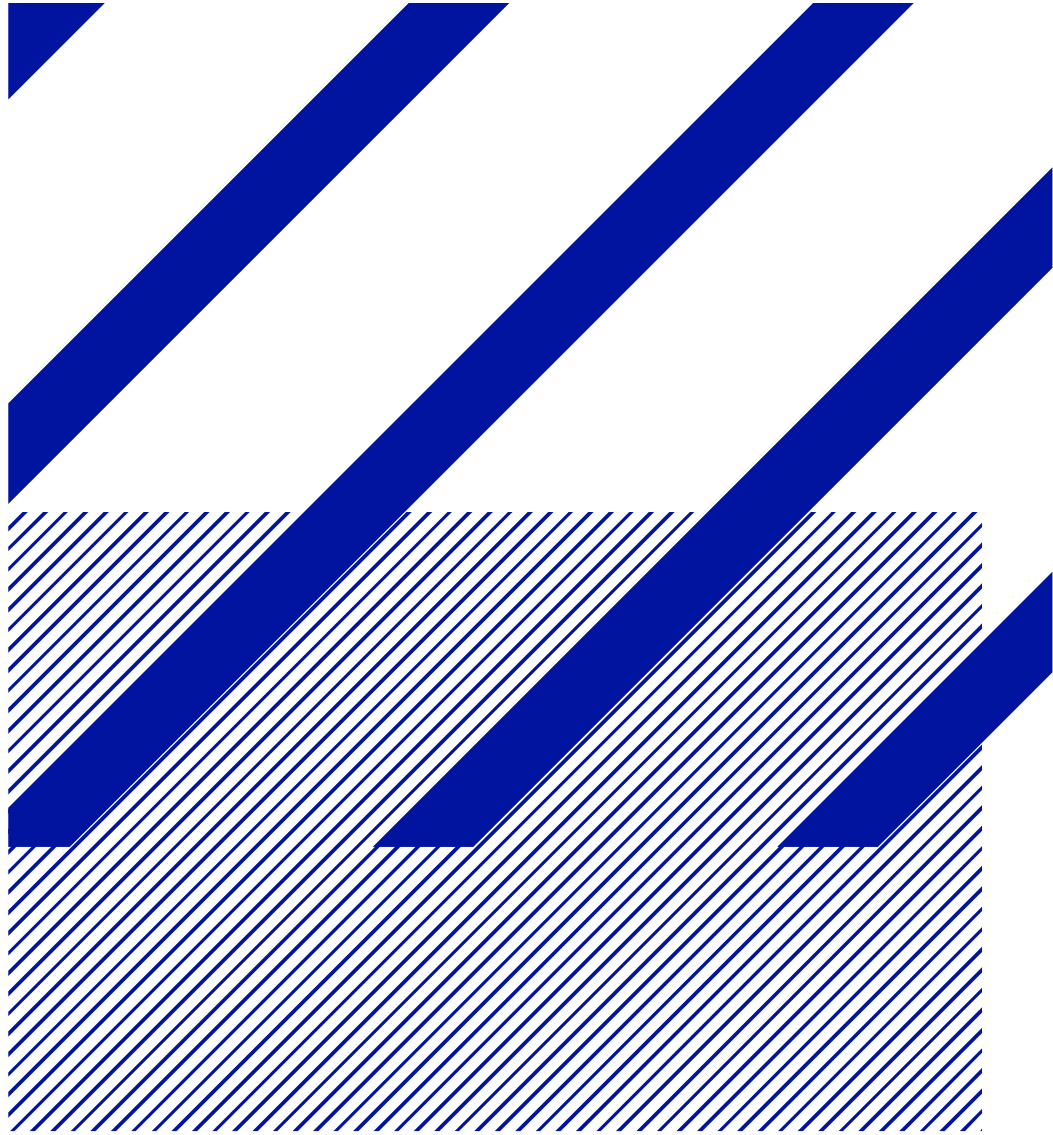
- Gegenmaßnahmen
  - WebRTC deaktivieren
  - Javascript deaktivieren
- Sicherheit oder Benutzerfreundlichkeit und Funktionsumfang

Interessante Webseiten

- Webseite aus dem aktuellen Angriff mit WebRTC und Javascript
  - `http://fb02itsscan06.fh-muenster.de/fingerprint`
- Andere Informationen die mittels Javascript ausgelesen werden können
  - `http://webkay.robinlinus.com/`
- Informationen über Facebook mittels Javascript und Timing Attack
  - `https://quadhead.de/timing-attacks/`

FH MÜNSTER
University of Applied Sciences

Vielen Dank für Ihre Aufmerksamkeit!



Jan Hölscher
Fachinformatiker für Systemintegration

E-Mail: hoelscher.jan@fh-muenster.de
Twitter: @wers_er_denn

Philipp Schwarz
Fachinformatiker für Anwendungsentwicklung

E-Mail: philipp.schwarz@fh-muenster.de
Twitter: @hawki1962