

# Network Fingerprinting for Securing User Accounts

Opportunities and Challenges

~ whoami

- Stephan Pinto Spindler
- Leading the Security Engineering team at 1&1 Mail & Media (web.de, GMX, mail.com)

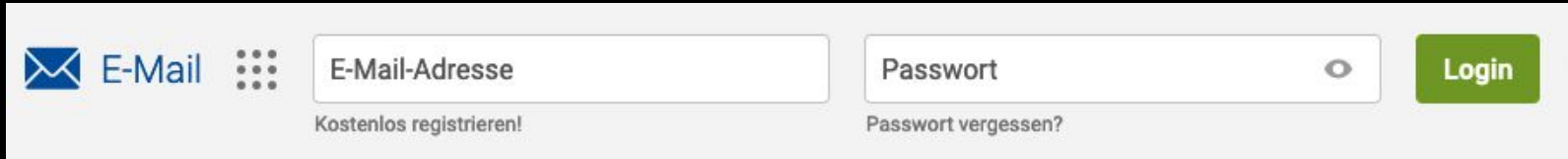


<https://github.com/s-spindler/>

<https://www.linkedin.com/in/stephan-pinto-spindler-61a511b2/>

# Protecting Users

# How can you protect your users?



The image shows a login form with the following elements:

- Logo: A blue envelope icon followed by the text "E-Mail" and a 3x3 grid of dots.
- Input field 1: A white box with the placeholder text "E-Mail-Adresse". Below it is the link "Kostenlos registrieren!".
- Input field 2: A white box with the placeholder text "Passwort" and a toggle eye icon on the right. Below it is the link "Passwort vergessen?".
- Button: A green button with the text "Login".



- Credential stuffing
- Password spraying
- Brute forcing

# How can you protect your users?

## **Rate limiting and (temporary) locks/bans!**

... but on what?

### **On accounts**

> Only works for targeted attacks

### **On source IPs**

> VPNs, NAT, distributed attacks

### **Do you even notice the attack?**

Metadata

# Looking at metadata

*method*  
GET / HTTP/1.1

*version*  
Host: web.de

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)...

Accept: text/html,application/xhtml+xml,...

Accept-Encoding: gzip, deflate, br

Accept-Language: en-US,en;q=0.9,de-DE;q=0.8

Referer: https://www.google.com/

Connection: keep-alive

Cookie: sessionId=123456789; userid=987654321...

*headers*

*cookies*

Transport Layer Security

▼ TLSv1.3 Record Layer: Handshake Protocol: Client Hello

Content Type: Handshake (22)

Version: TLS 1.0 (0x0301)

Length: 649

▼ Handshake Protocol: Client Hello

Handshake Type: Client Hello (1)

Length: 645

> Version: TLS 1.2 (0x0303)

Random: 56c5e19d41daf2c2b5f3a066fe5465580d2cbf8013f22abc49aa94a831c86722

Session ID Length: 32

Session ID: b07556f3340e00c2776a0a706ec2277f8cb877f70afc26e22bd56911420a7193

Cipher Suites Length: 34

> Cipher Suites (17 suites)

Compression Methods Length: 1

> Compression Methods (1 method)

Extensions Length: 538

> Extension: server\_name (len=11) name=web.de

> Extension: extended\_master\_secret (len=0)

> Extension: renegotiation\_info (len=1)

> Extension: supported\_groups (len=14)

> Extension: ec\_point\_formats (len=2)

> Extension: session\_ticket (len=0)

> Extension: application\_layer\_protocol\_negotiation (len=14)

> Extension: status\_request (len=5)

> Extension: delegated\_credentials (len=10)

> Extension: key\_share (len=107) x25519, secp256r1

> Extension: supported\_versions (len=5) TLS 1.3, TLS 1.2

> Extension: signature\_algorithms (len=24)

> Extension: psk\_key\_exchange\_modes (len=2)

> Extension: record\_size\_limit (len=2)

> Extension: encrypted\_client\_hello (len=281)



## There's more...

- TCP/QUIC metadata: duration between steps in the handshake
- HTTP2: settings for, e.g., frame sizes

Focus today: HTTP and TLS

# Fingerprints

# Common fingerprint "standards" - JA3

- **JA3:** invented at Salesforce but no longer maintained
- Still supported in various tools
- Uses these fields from Client Hello:

SSLVersion, Cipher, SSLExtension, EllipticCurve, EllipticCurvePointFormat

```
[JA3 Fullstring: 771,4865-4867-4866-49195-49199-52393-52392-49196-49200-49197-49198-49199-49200-49201-49202-49203-49204-49205-49206-49207-49208-49209-49210-49211-49212-49213-49214-49215-49216-49217-49218-49219-49220-49221-49222-49223-49224-49225-49226-49227-49228-49229-49230-49231-49232-49233-49234-49235-49236-49237-49238-49239-49240-49241-49242-49243-49244-49245-49246-49247-49248-49249-49250-49251-49252-49253-49254-49255-49256-49257-49258-49259-49260-49261-49262-49263-49264-49265-49266-49267-49268-49269-49270-49271-49272-49273-49274-49275-49276-49277-49278-49279-49280-49281-49282-49283-49284-49285-49286-49287-49288-49289-49290-49291-49292-49293-49294-49295-49296-49297-49298-49299-49300-49301-49302-49303-49304-49305-49306-49307-49308-49309-49310-49311-49312-49313-49314-49315-49316-49317-49318-49319-49320-49321-49322-49323-49324-49325-49326-49327-49328-49329-49330-49331-49332-49333-49334-49335-49336-49337-49338-49339-49340-49341-49342-49343-49344-49345-49346-49347-49348-49349-49350-49351-49352-49353-49354-49355-49356-49357-49358-49359-49360-49361-49362-49363-49364-49365-49366-49367-49368-49369-49370-49371-49372-49373-49374-49375-49376-49377-49378-49379-49380-49381-49382-49383-49384-49385-49386-49387-49388-49389-49390-49391-49392-49393-49394-49395-49396-49397-49398-49399-49400-49401-49402-49403-49404-49405-49406-49407-49408-49409-49410-49411-49412-49413-49414-49415-49416-49417-49418-49419-49420-49421-49422-49423-49424-49425-49426-49427-49428-49429-49430-49431-49432-49433-49434-49435-49436-49437-49438-49439-49440-49441-49442-49443-49444-49445-49446-49447-49448-49449-49450-49451-49452-49453-49454-49455-49456-49457-49458-49459-49460-49461-49462-49463-49464-49465-49466-49467-49468-49469-49470-49471-49472-49473-49474-49475-49476-49477-49478-49479-49480-49481-49482-49483-49484-49485-49486-49487-49488-49489-49490-49491-49492-49493-49494-49495-49496-49497-49498-49499-49500-49501-49502-49503-49504-49505-49506-49507-49508-49509-49510-49511-49512-49513-49514-49515-49516-49517-49518-49519-49520-49521-49522-49523-49524-49525-49526-49527-49528-49529-49530-49531-49532-49533-49534-49535-49536-49537-49538-49539-49540-49541-49542-49543-49544-49545-49546-49547-49548-49549-49550-49551-49552-49553-49554-49555-49556-49557-49558-49559-49560-49561-49562-49563-49564-49565-49566-49567-49568-49569-49570-49571-49572-49573-49574-49575-49576-49577-49578-49579-49580-49581-49582-49583-49584-49585-49586-49587-49588-49589-49590-49591-49592-49593-49594-49595-49596-49597-49598-49599-49600-49601-49602-49603-49604-49605-49606-49607-49608-49609-49610-49611-49612-49613-49614-49615-49616-49617-49618-49619-49620-49621-49622-49623-49624-49625-49626-49627-49628-49629-49630-49631-49632-49633-49634-49635-49636-49637-49638-49639-49640-49641-49642-49643-49644-49645-49646-49647-49648-49649-49650-49651-49652-49653-49654-49655-49656-49657-49658-49659-49660-49661-49662-49663-49664-49665-49666-49667-49668-49669-49670-49671-49672-49673-49674-49675-49676-49677-49678-49679-49680-49681-49682-49683-49684-49685-49686-49687-49688-49689-49690-49691-49692-49693-49694-49695-49696-49697-49698-49699-49700-49701-49702-49703-49704-49705-49706-49707-49708-49709-49710-49711-49712-49713-49714-49715-49716-49717-49718-49719-49720-49721-49722-49723-49724-49725-49726-49727-49728-49729-49730-49731-49732-49733-49734-49735-49736-49737-49738-49739-49740-49741-49742-49743-49744-49745-49746-49747-49748-49749-49750-49751-49752-49753-49754-49755-49756-49757-49758-49759-49760-49761-49762-49763-49764-49765-49766-49767-49768-49769-49770-49771-49772-49773-49774-49775-49776-49777-49778-49779-49780-49781-49782-49783-49784-49785-49786-49787-49788-49789-49790-49791-49792-49793-49794-49795-49796-49797-49798-49799-49800-49801-49802-49803-49804-49805-49806-49807-49808-49809-49810-49811-49812-49813-49814-49815-49816-49817-49818-49819-49820-49821-49822-49823-49824-49825-49826-49827-49828-49829-49830-49831-49832-49833-49834-49835-49836-49837-49838-49839-49840-49841-49842-49843-49844-49845-49846-49847-49848-49849-49850-49851-49852-49853-49854-49855-49856-49857-49858-49859-49860-49861-49862-49863-49864-49865-49866-49867-49868-49869-49870-49871-49872-49873-49874-49875-49876-49877-49878-49879-49880-49881-49882-49883-49884-49885-49886-49887-49888-49889-49890-49891-49892-49893-49894-49895-49896-49897-49898-49899-49900-49901-49902-49903-49904-49905-49906-49907-49908-49909-49910-49911-49912-49913-49914-49915-49916-49917-49918-49919-49920-49921-49922-49923-49924-49925-49926-49927-49928-49929-49930-49931-49932-49933-49934-49935-49936-49937-49938-49939-49940-49941-49942-49943-49944-49945-49946-49947-49948-49949-49950-49951-49952-49953-49954-49955-49956-49957-49958-49959-49960-49961-49962-49963-49964-49965-49966-49967-49968-49969-49970-49971-49972-49973-49974-49975-49976-49977-49978-49979-49980-49981-49982-49983-49984-49985-49986-49987-49988-49989-49990-49991-49992-49993-49994-49995-49996-49997-49998-49999-50000]
```



screenshot straight out of Wireshark

# Common fingerprint "standards" - JA4

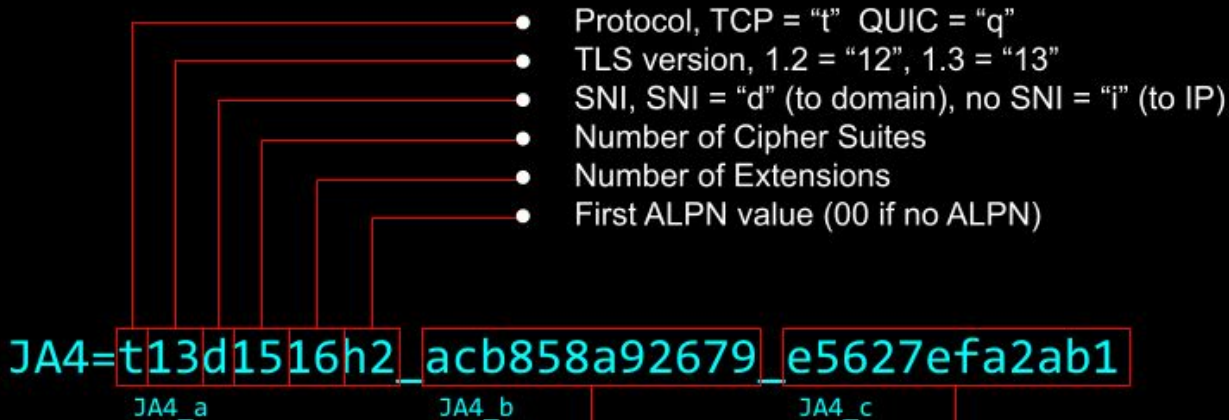
**JA4+ suite:** by FoxIO, builds on JA3, adds some more

- Methods for TLS, HTTP, TCP, SSH
- Both client and server

Licensing note:

- Only TLS client fingerprinting (JA4) is BSD licensed
- The rest: FoxIO license - still good to go for internal business use

## JA4: TLS Client Fingerprint




- Truncated SHA256 hash of the Cipher Suites, sorted
- Truncated SHA256 hash of the Extensions, sorted + Signature Algorithms, in the order they appear

# JA4H: HTTP Client Fingerprint

- HTTP Method, GET = "ge", PUT = "pu", POST = "po", etc
- HTTP Version, 2.0 = "20", 1.1 = "11"
- Cookie, if there's a Cookie "c", if no Cookie "n"
- Referer, if there's a Referer "r" if no Referer "n"
- Number of HTTP Headers (ignoring Cookie and Referer)
- First 4 characters of primary Accept-Language (0000 if no Accept-Language)

JA4H=ge20cr13enus\_974ebe531c03\_b66fa821d02c\_e97928733c74

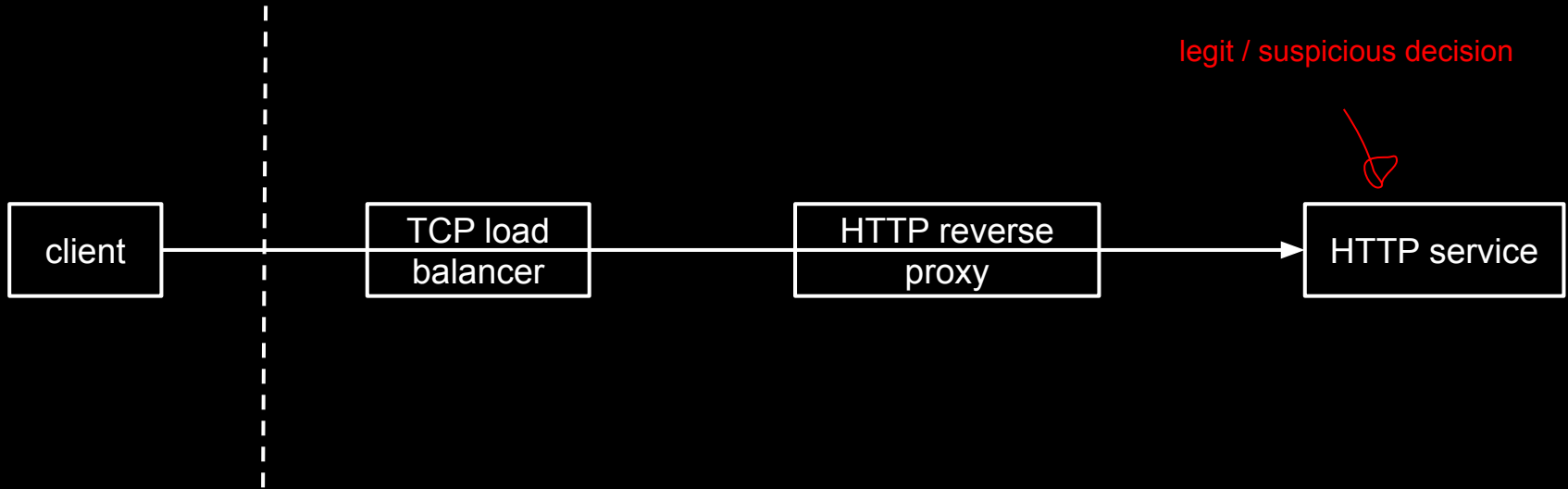
JA4H\_a                      JA4H\_b                      JA4H\_c                      JA4H\_d



- Truncated SHA256 hash of Headers, in the order they appear
- Truncated SHA256 hash of Cookie Fields, sorted
- Truncated SHA256 hash of Cookie Fields + Values, sorted

# Technical Challenges

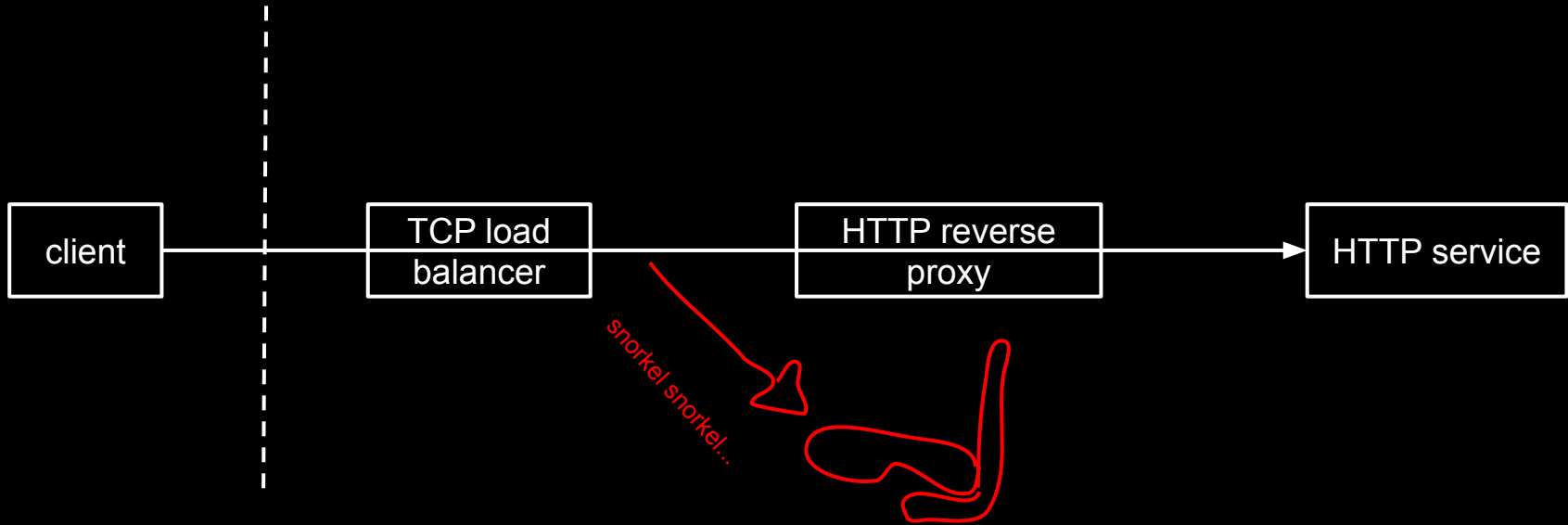
# Sample setup



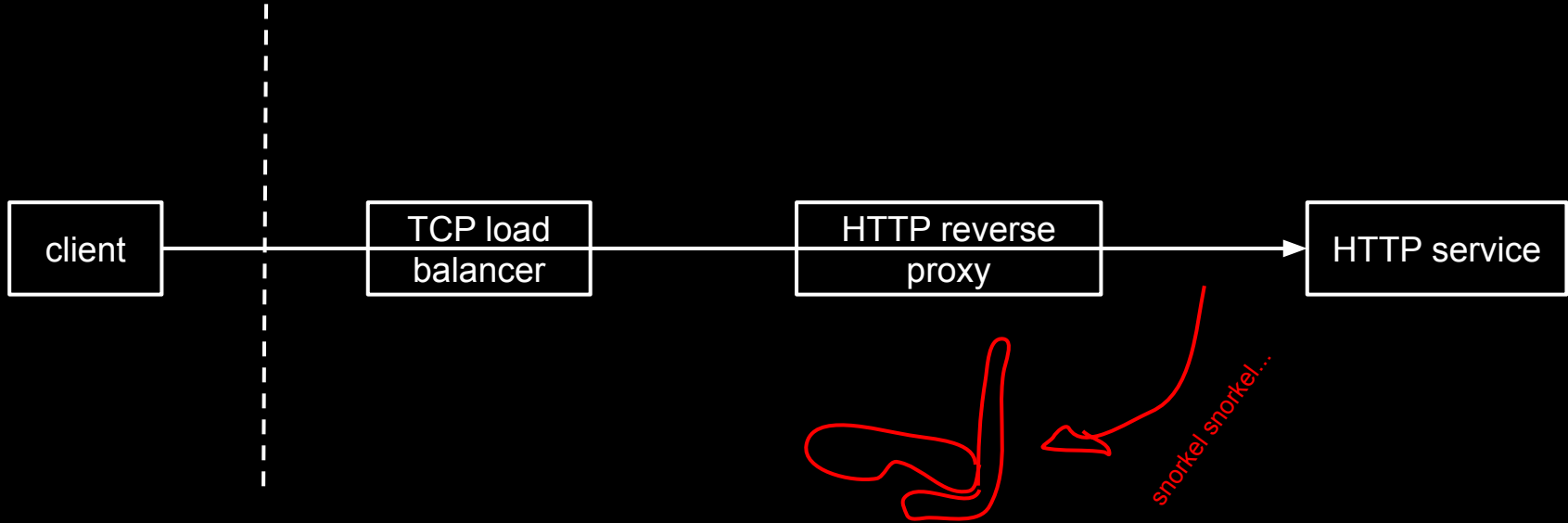


# Out-of-Band Fingerprint Creation

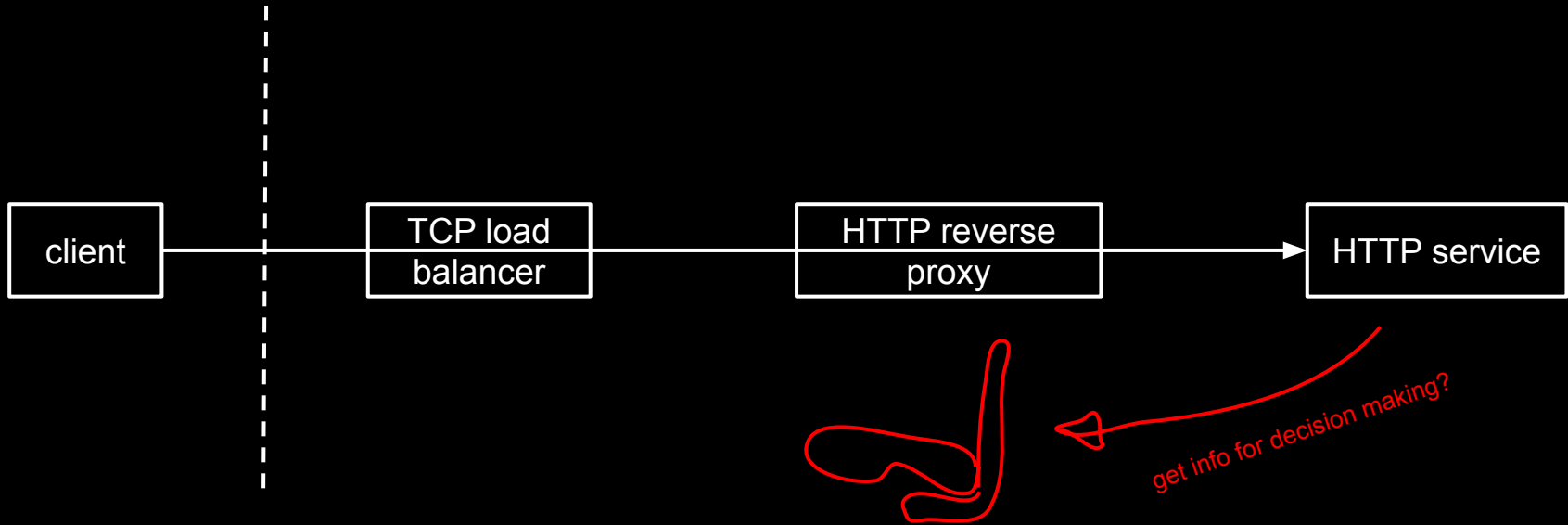
# Challenges for out-of-band fingerprint creation



# Challenges for out-of-band fingerprint creation

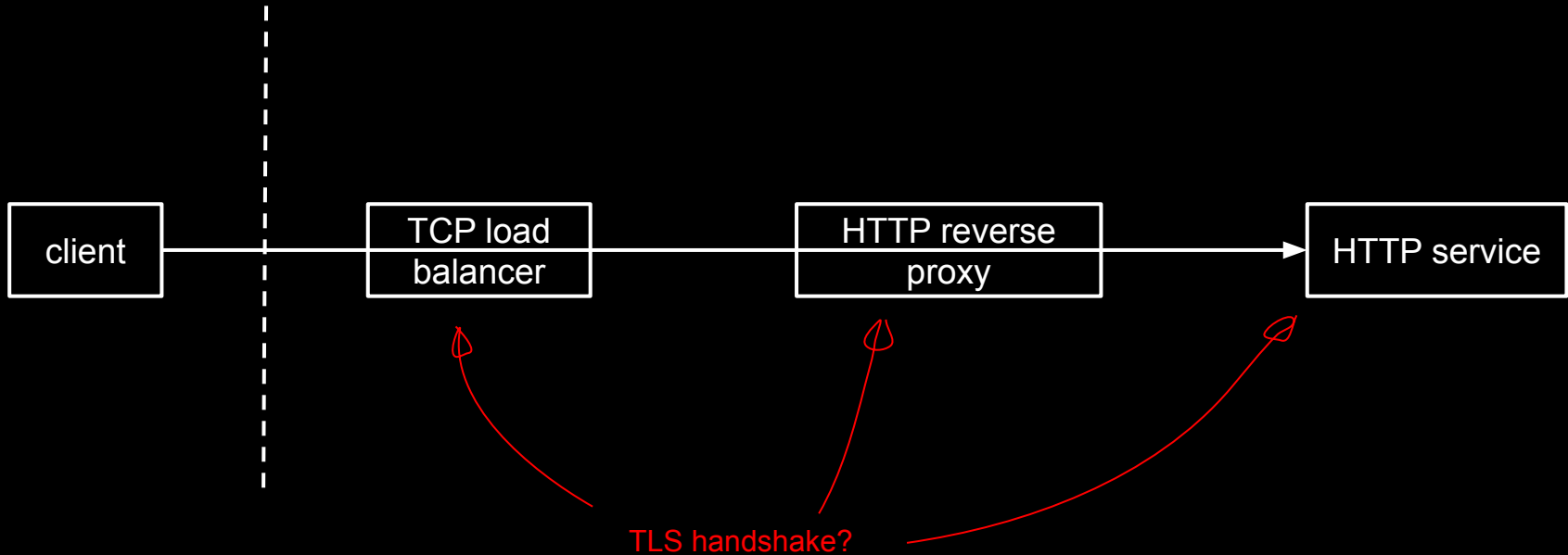


# Challenges for out-of-band fingerprint creation

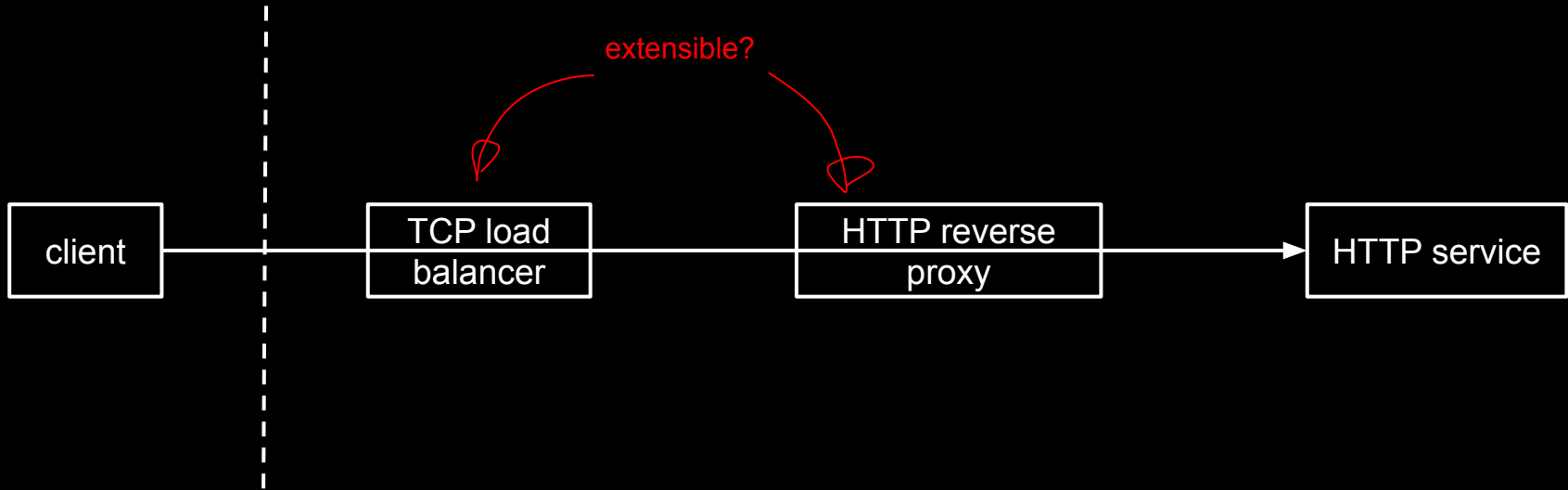


# In-Stream Fingerprint Creation

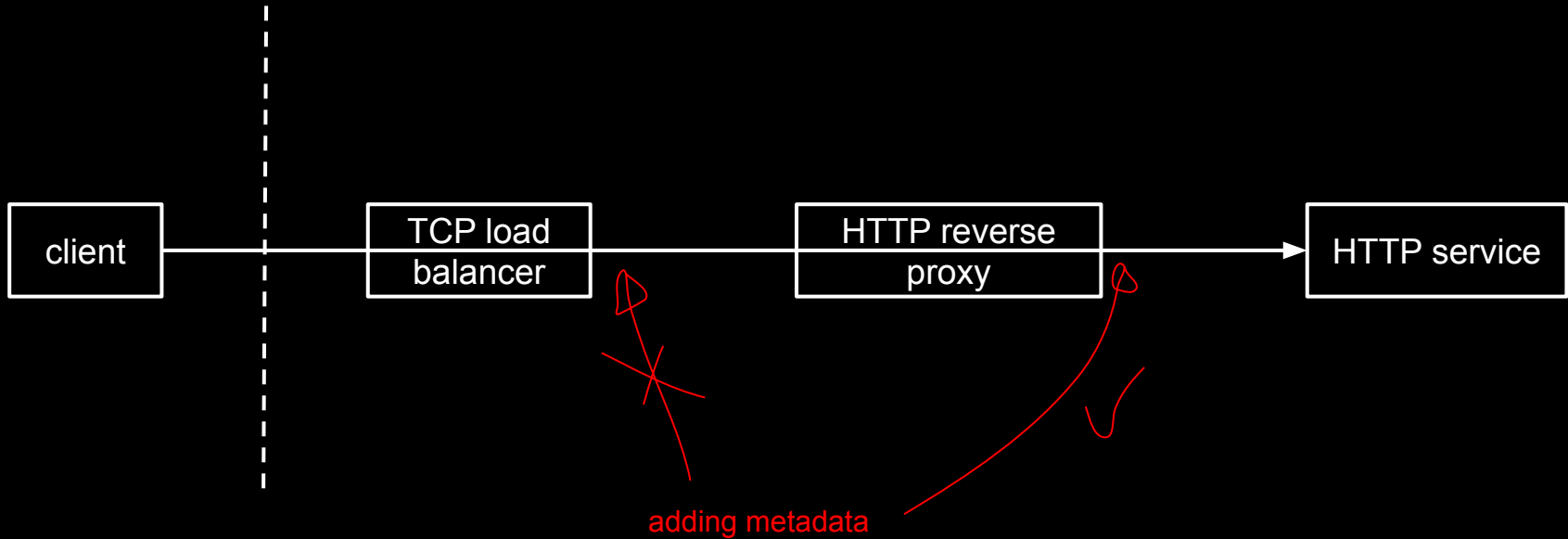
# Challenges for in-stream fingerprint creation



# Challenges for in-stream fingerprint creation



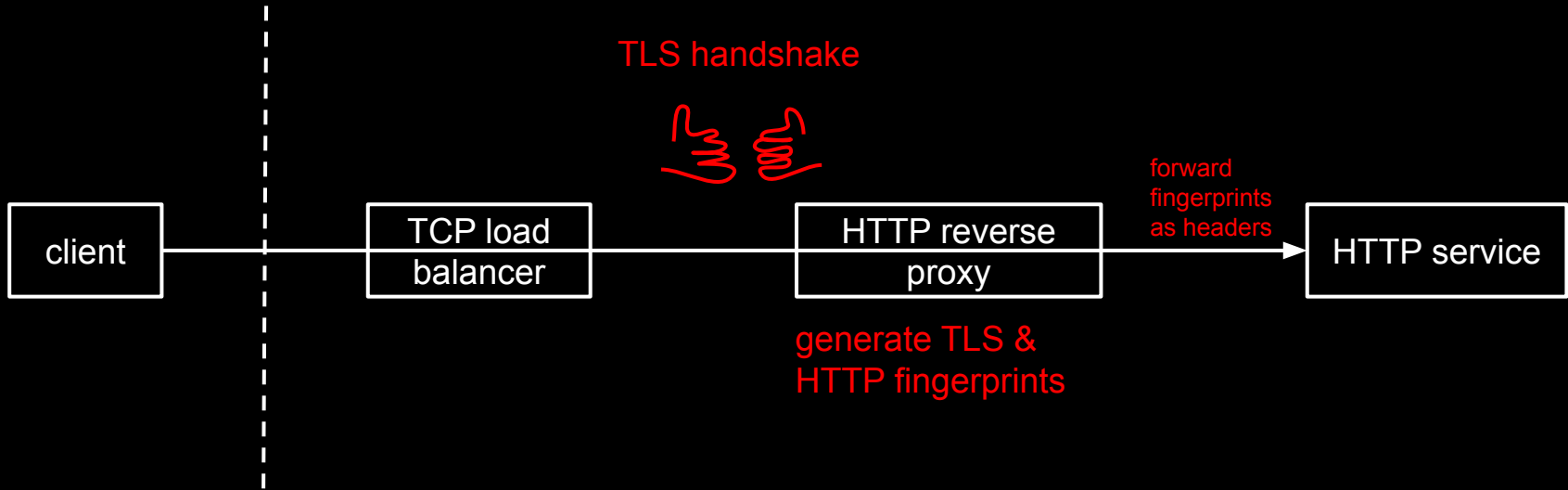
# Challenges for in-stream fingerprint creation





# Possible setup for in-stream creation

TLS offloading in reverse proxy (can add TLS on upstream connection again)



# Implementing HTTP fingerprinting - traefik plugin example

```
func (p *Plugin) ServeHTTP(rw http.ResponseWriter, req *http.Request) {  
  
    method := req.Method  
  
    version := fmt.Sprintf("%d%d", req.ProtoMajor, req.ProtoMinor)  
  
    cookies := req.Cookies()  
  
    referer := req.Referer()  
  
    headers := req.Header  
  
    lang := req.Header.Get("Accept-Language")  
  
    req.Header.Add("X-FP-HTTP", createFingerprint(method, version, cookies, referer, headers, lang))  
  
    p.next.ServeHTTP(rw, req)  
}
```

# Manipulating HTTP metadata - Go example

```
client := &http.Client{
    Transport: &http.Transport{ forces HTTP 1.1
        TLSNextProto: map[string]func(string, *tls.Conn) http.RoundTripper{},
    },
}
```

```
req, err := http.NewRequest("GET", "https://web.de", nil)
if err != nil {
    panic(err)
}
```

```
req.AddCookie(&http.Cookie{Name: "mycookie", Value: "blabla"})
req.Header.Set("Custom-Header", "some value")
req.Header.Set("User-Agent", "Chrome/123.4.5.67")
```

**manipulate headers and cookies**

```
resp, err := client.Do(req)
```

# Manipulating HTTP metadata - curl example

```
curl https://web.de \  
  --http1.1 \  
  --user-agent "Chrome/123.4.5.67" \  
  --cookie "mycookie=blabla" \  
  --header "Custom-Header: some value"
```

# Implementing TLS fingerprinting - traefik plugin example

```
func (p *Plugin) ServeHTTP(rw http.ResponseWriter, req *http.Request) {  
  
    negotiated_version := req.TLS.Version  
  
    sni := req.TLS.ServerName  
  
    alpn := req.TLS.NegotiatedProtocol  
  
    negotiated_cipher := req.TLS.CipherSuite  
  
    req.Header.Add("X-FP-TLS", createFingerprint(negotiated_version, sni, alpn, negotiated_cipher))  
  
    p.next.ServeHTTP(rw, req)  
}
```

> Missing TLS handshake information (supported versions, cipher suites, extensions)

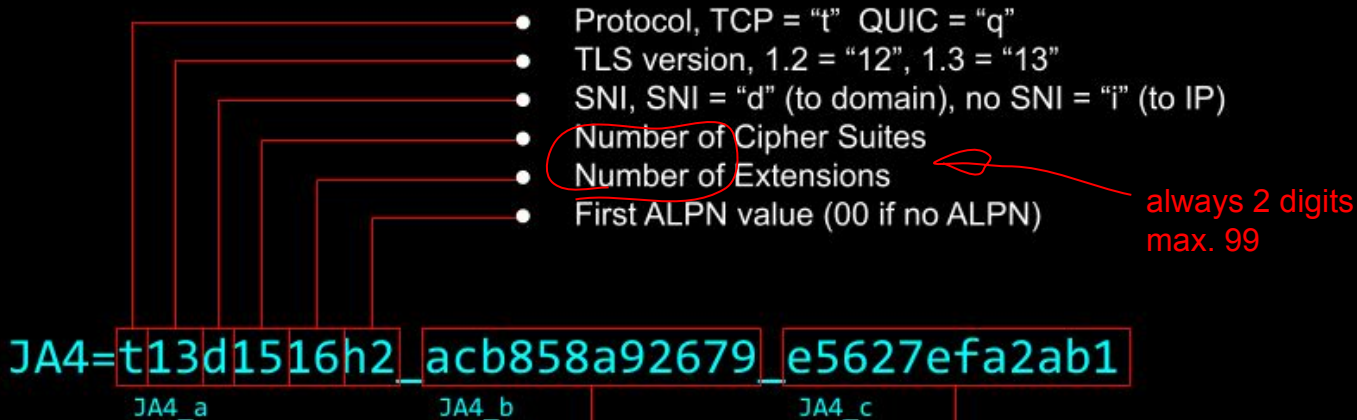
# Manipulating TLS metadata - curl example

```
curl https://httpbin.ams.fe-intg-iz2-bap.poinfra.server.lan/anything \  
  --alpn \  
  --tlsv1.2 \    sets minimum TLS version to 1.2  
  --tls-max "1.3" \  
  --ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384 \ specifies ciphers for TLS versions <= 1.2  
  --tls13-ciphers TLS_AES_128_GCM_SHA256
```

> No easy way of manipulating extensions

# Technical Challenges - Bonus Round

## JA4: TLS Client Fingerprint

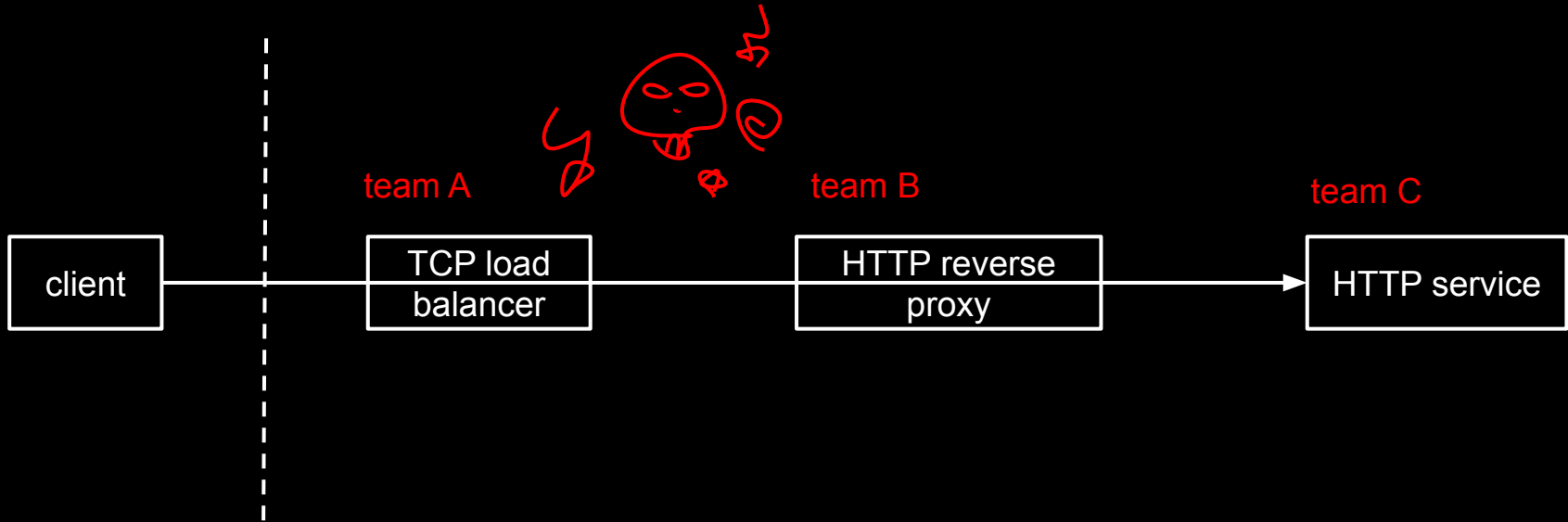


- Truncated SHA256 hash of the Cipher Suites, sorted
- Truncated SHA256 hash of the Extensions, sorted + Signature Algorithms, in the order they appear



# Organizational Challenges

# Organizational challenges



# Regulatory Challenges

# Privacy

- Fingerprints can count as personally identifiable information
- Storage might be off-limits
- Talk to your data privacy officer



Conclusion

# Still worth the effort?

- Probably yes
- Avoid being an easy target
- Drive up cost for attackers

Thank you!