Proximax, Telex, Flashproxy

The current state of circumvention software

Jens Kubieziel
<jens@kubieziel.de>

29th Chaos Communication Congress

January 2, 2013

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

What this talk will be about

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Censorship

Censorship is an old and worldwide problem.

- My university had an own censorship authority several hundred years ago.
- Germany has no censorship and this is stated in the Grundgesetz:

  Eine Zensur findet nicht statt. Art. 5 Abs. 1 Grundgesetz

- Other countries still try to block the flow of information.

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Censorship worldwide

Examples of censorship

- Cleanfeed in the UK
- blocking or modifying of emails in Libya
- fake websites in Kazakhstan
- and of course the Great Firewall of China

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Censorship as danger to the people

Censorship is not anymore a means to block information, but the tools are used to track people down, to torture and even to murder them.

Fact

We need secure, blocking-resistance ways to communicate, especially for activists.

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

What this talk will be about

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Existing ways of circumvention

Some ways are quite non-technical:

- no `www`
- use HTTPS
- change the case of the domain name (e. g. `http://ExAmplE.org/`)
- encode URLs (e. g. `http://example.org/index%2Ehtml`)

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Existing software for circumvention

Open ProxiesOpen Proxies	VPNsVPNs	Alkasir
PsiphonPsiphon	Your Freedom	Collage
InfranetInfranet	Tangler	Triangle Boy
Freehaven	UltrasurfUltrasurf	SWEET
Cirripede	ProximaxProximax	DynaWeb
SafewebSafeweb	HaystackHaystack	Peek-a-booty
TelexTelex	JonDonym	Censorswee
Instasurf	Hotspot Shield	WebSecure
TorTor	FlashproxiesFlashproxies	BridgesBridg
Picidae	Message in a bottle	#h00t

What this talk will be about

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Attacker model

In general, we talk about an adversary who can

- log network packets
- mount active attacks (inject packets, modify packets etc.)

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Infranet

Overview

- proposed in 2002 by Nick Feamster et al.
- builds a covert tunnel between a *requester* and *responder*
- sends HTTP messages back and forth

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Infranet

Design goals

1. (statistical) deniability for the requester
2. covertness for the responder
3. robustness of communication
4. performance

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Infranet

Protocol

Definition

Requester and responder send HTTP messages which is treated with a *hiding function* $\mathcal{H}(m, c, s)$, where m is a message, c a cover and s a secret.

Infranet makes a distinction between up- and downstream communication. Upstream consists of different URLs (or HTTP, TCP headers) and downstream consists of JPG images.

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Infranet

Tunnel

- Tunnel setup
- upstream communication
- downstream communication

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Infranet

Tunnel setup

1. requester makes initial connection (`index.html`)
2. responder creates a unique ID and sets it via URL manipulation or cookie
3. requester sends $\mathcal{H}(U_{\text{init}}, \text{HTTP Request}, s)$
4. responder sends $\mathcal{H}(U_{\text{tunnel}}, \text{HTTP Response}, s)$
5. both send `Transmit Request` and `Transmit Response`

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Infranet

Upstream communication

The requester divides a message into several parts and sends each as single HTTP request. The responder uses its information to recover the message.

- implicit mapping
- based on a dictionary

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Infranet

Downstream communication

The messages are hidden inside steganographic images.

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Infranet

Security

- Discovery attacks
- Replay attacks
- Addition or deletion attacks
- Selective degradation

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Infranet

Selective degradation

A censor does a deletion attack with probability p and correctly forwards with $1-p$. Download time for normal users increases a bit, but Infranet has to reinitialize.

- expected number of requests for a normal user: $\frac{1}{1-p}$
- expected number of requests for an Infranet user, who issues n requests: $\frac{n}{(1-p)^n}$

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Proximax

- proposed by Damon McCoy et al.
- assembles a large pool of proxies
- distributes them so that the usage is maximized

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Proximax

Design

The design of Proximax relies on the users. They learn about proxies from Proximax and distribute them.

Distinction between

- registered users and
- normal users

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Proximax

Design

Proximax tracks the

- usage rate and
- risk of being blocked

Measure: number of user-hours a proxy provides, *yield*.

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Proximax

Three main tasks of operation

1. Administrators, who run proxies
2. Managing channels
3. Inviting users

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Proximax

Modeling the system

m number of resources (proxies)

n number of disseminating channels (users)

R_i set of resources advertised via channel i

t_i Resource i lifetime

λ_j Channel j risk

u_j Usage of channel j

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Proximax

Modeling the system

The total risk and total usage of resource i can be written as

$$\Lambda_i = \gamma + \sum_{j \in A_i} \lambda_j \qquad U_i = \sum_{j \in A_i} u_j$$

where A_i is a set of channels which advertise a resource.
So the expected yield of a resource is

$$\frac{U_i}{\Lambda_i}$$

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Proximax

Maximum likelihood estimate

We can use the log-likelihood function:

$$\ell = \log \prod_{i=1}^{m} \Lambda_i e^{-\Lambda_i t_i} = \log(\Lambda_1 e^{-\Lambda_1 t_1} \cdot \ldots \cdot \Lambda_m e^{-\Lambda_m t_m})$$

$$= \log(\Lambda_1) - \Lambda_1 t_1 + \cdots + \log(\Lambda_m) - \Lambda_m t_m$$

$$= \sum_{i=1}^{m} (\log \Lambda_i - \Lambda_i t_i)$$

$$\frac{\partial \ell}{\partial \lambda_j} = \sum \left(\frac{1}{\Lambda_i} - t_i \right)$$

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Proximax

Maximizing the total yield

Basically the resource $\tilde{j}$ is chosen which maximises the yield:

$$\Delta_i = \frac{u_{\tilde{j}} + U_i}{\lambda_{\tilde{j}} + \Lambda_i} - \frac{U_i}{\Lambda_i}$$

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Proximax

Some possible attacks

- pwn the administrators
- censor shares its data
- increase yield and block

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

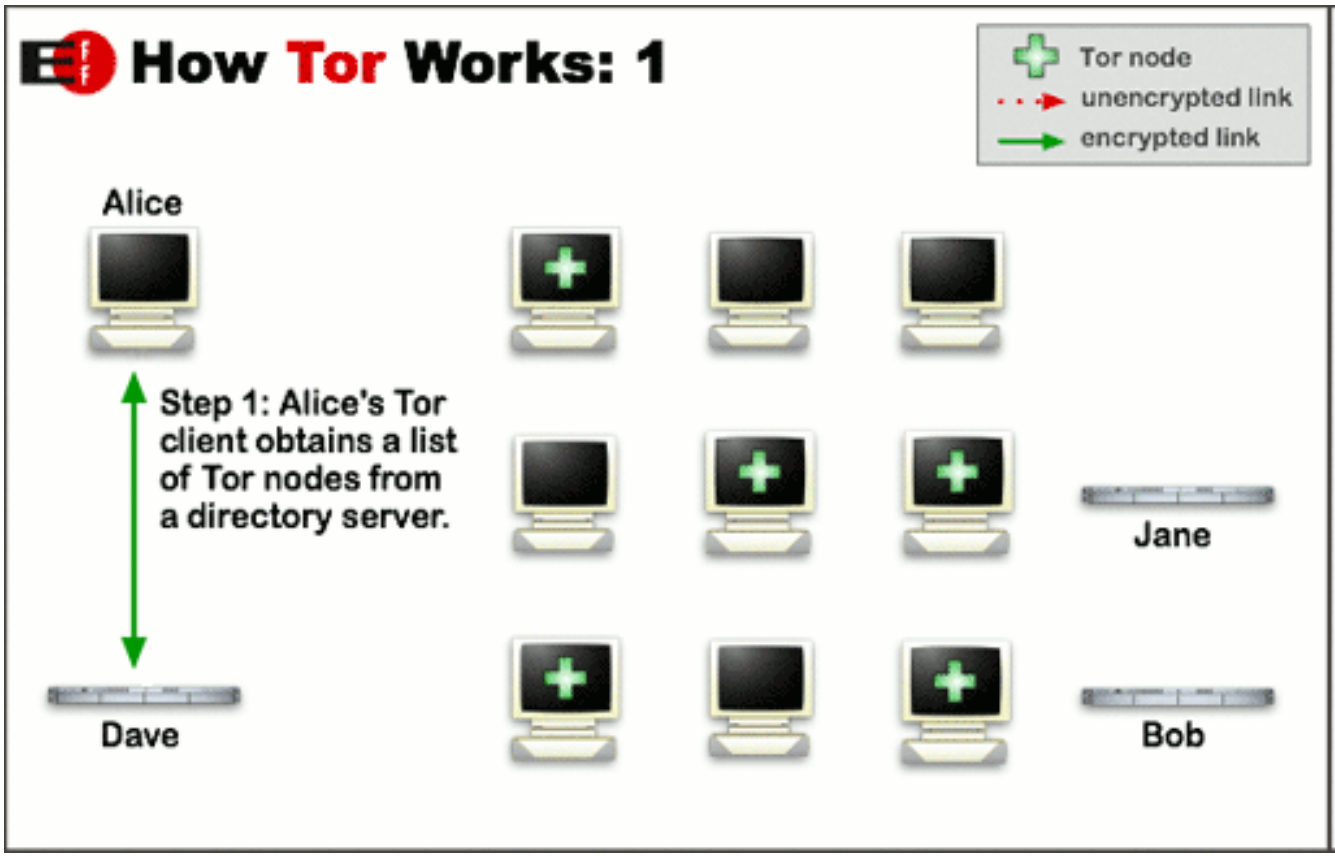
Tor

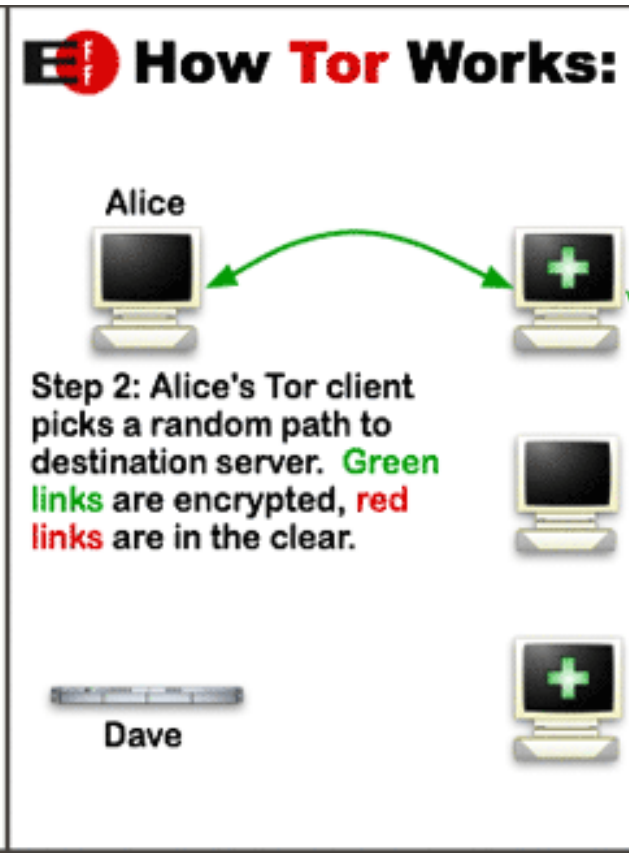
- proposed by Roger Dingledine et al.
- one of the most used and well researched anonymity software in the wild
- research into circumvention

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Tor

How Tor works





N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Tor Bridges

Tor bridges are literally a bridge into the Tor network. Contrary to all relays in directory authorities the entries in the bridge authority are “hidden”. Bridges usually are distributed

- in a private manner
- through the site `http://bridges.torproject.org/`
- via (e|G)mail to `bridges@torproject.org`
- by asking guys from TorProject.org

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

obfsproxy

The job of obfsproxy is to obscure the traffic between a client and a Tor bridge (framework). It is based on a plugin architecture. Plugins can simulate several kinds of traffic (HTTPS, StegoTorus, Skype Video etc.)

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

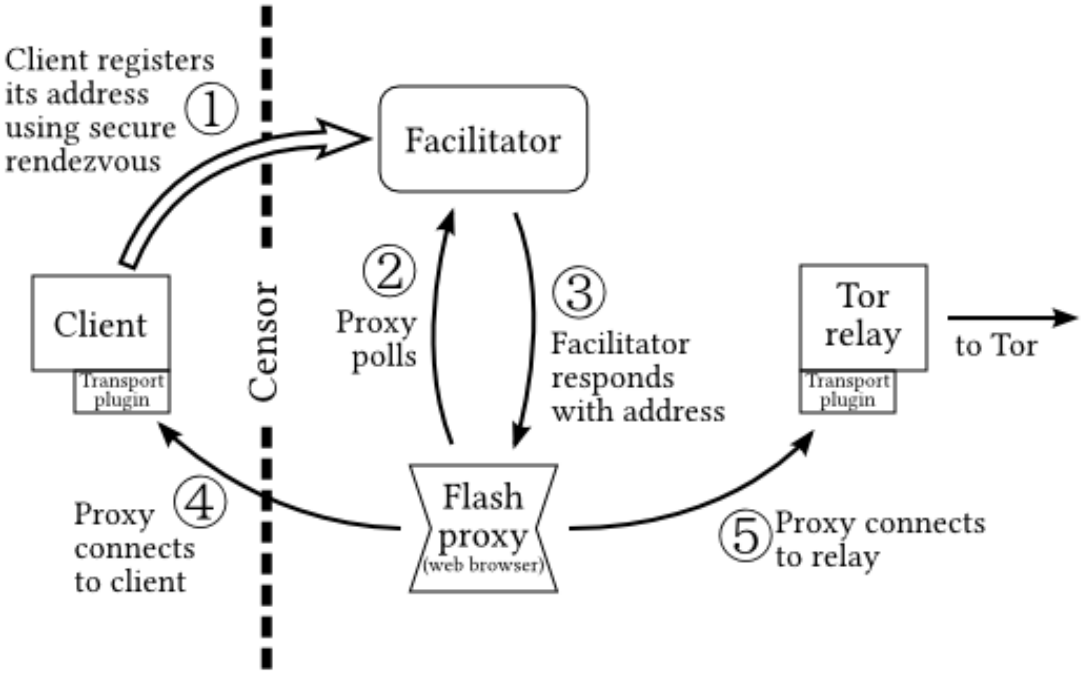
Flash proxies

The flash proxy system uses browsers all over the Internet as ephemeral proxies.
David Fifield *et al.: Evading Censorship with Browser-Based Proxies*

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Flash proxies

Architecture



(image from https://crypto.stanford.edu/flashproxy/)

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Flash proxies

Limitations

In-browser software can't just open a socket and wait for connections.

It has to open outside connections.

Furthermore there are security policies at the browser side:

- WebSocket: Cross-Origin Resource Sharing (CORS), send HTTP-Header `Acess-Control-Allow-Origin`
- Flash: Endpoints must serve crossdomain policy

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.–30.12./
HA/M.B-U/RG

Telex

- proposed by Eric Wustrow et al.
- needs ISPs who install a Telex station
- Telex station looks for “tags” and does some steganographic and TLS magic

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Telex

Design Overview

1. Client select an unblocked website and connects to that site using HTTPS.
2. Telex client inserts a “tag” which looks like nonce (and is a reference to the blocked site)
3. ISP forwards the request to the Telex station
4. Telex station recognizes the tag and instructs the ISP router to forward all packets to the station
5. Telex station now diverts all traffic to the blocked site

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Telex

Tagging

A tag has to be *short* and *indistinguishable* from a random string.Telex uses

- a private key r
- a public key $\alpha = g^r$
- two cryptographically secure hash functions H_1 and H_2

To construct a tag:

1. client chooses a random key s
2. calculates g^s and $\alpha^s = g^{rs}$
3. The tag is $g^s \| H_1(g^{rs} \| \chi)$
4. The shared secret key is $H_2(g^{rs} \| \chi)$

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Telex

Handshake

Telex does some tweaked TLS handshake:

1. Client sends a `ClientHello` with tag as random value
2. Telex station observes the tag, extracts the nonce and learns the shared key
3. server does his part of initiating a TLS connection
4. clients seeds a PRG with shared secret and uses that value for key exchange
5. Telex station simulates the client and also gets the secret
6. Telex station takes over the TLS session and sends a RST to the original server

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG

Further development

- Cirripede
- SWEET
- CensorSweeper

N.O-T/M.Y-D
E/PA.R-TM/E
N.T-**2.9-C/3**
27.-30.12./
HA/M.B-U/RG