Abusing Transparent Proxies with Flash

v2.00

By Robert Auger
PayPal Information Risk Management

AppSec DC

November 2009

The OWASP Foundation

http://www.owasp.org

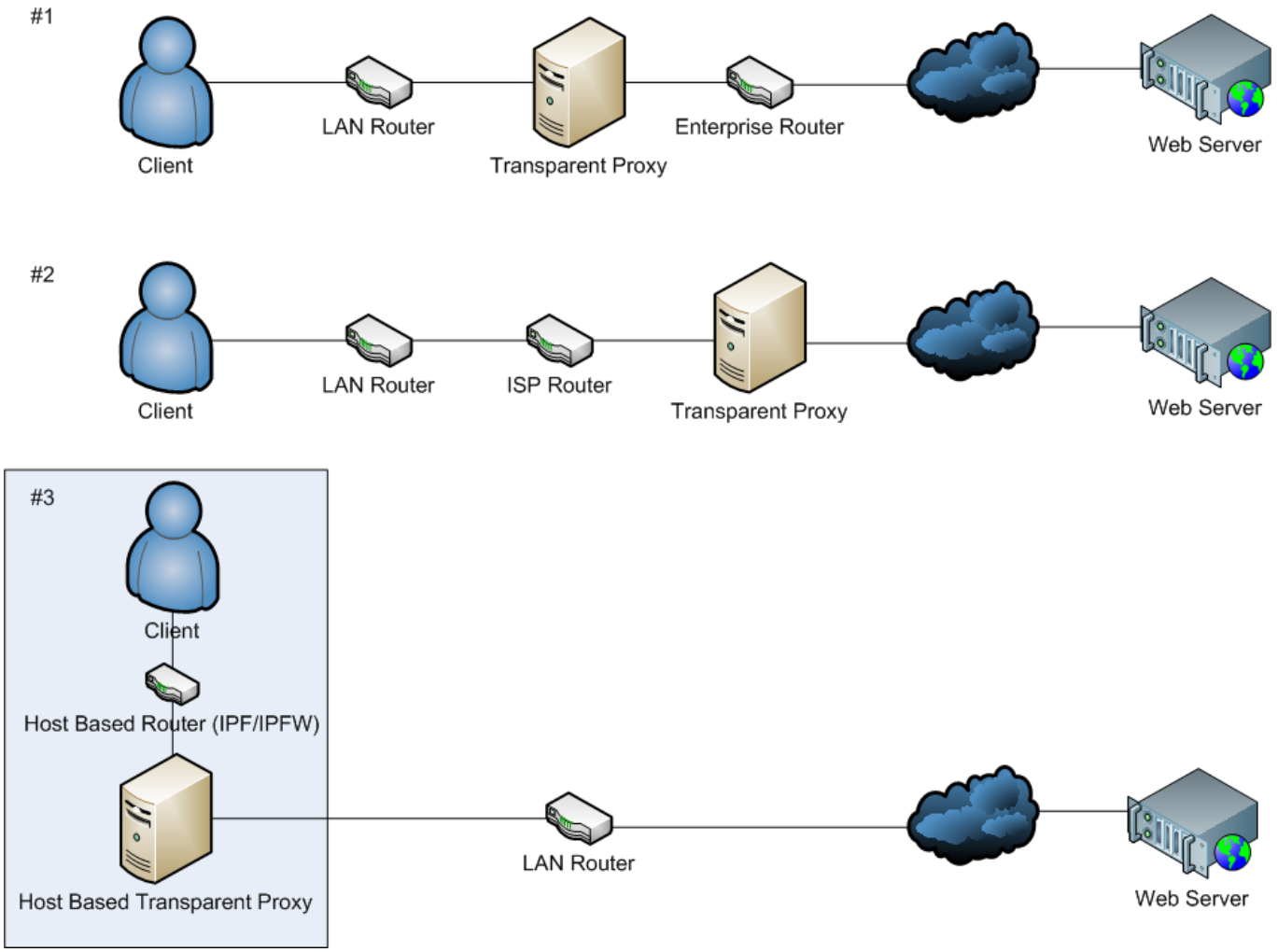
Overview

- What are transparent and intercepting proxies?
- When are transparent proxies used?
- How do they operate?
- Brief intro to the SOP
- Flash and the socket policy
- The abuse case
- Solutions and mitigations
- Conclusions

What are transparent and intercepting proxies?

- **Explicit Proxy**: A proxy explicitly configured by a client or user system. Also known as a classic web proxy.
- **Transparent Proxy**: Proxy which is NOT explicitly configured by the client machine.
- **Intercepting Proxy**: A more intrusive version of a transparent proxy. May modify traffic.

When are transparent proxies used?



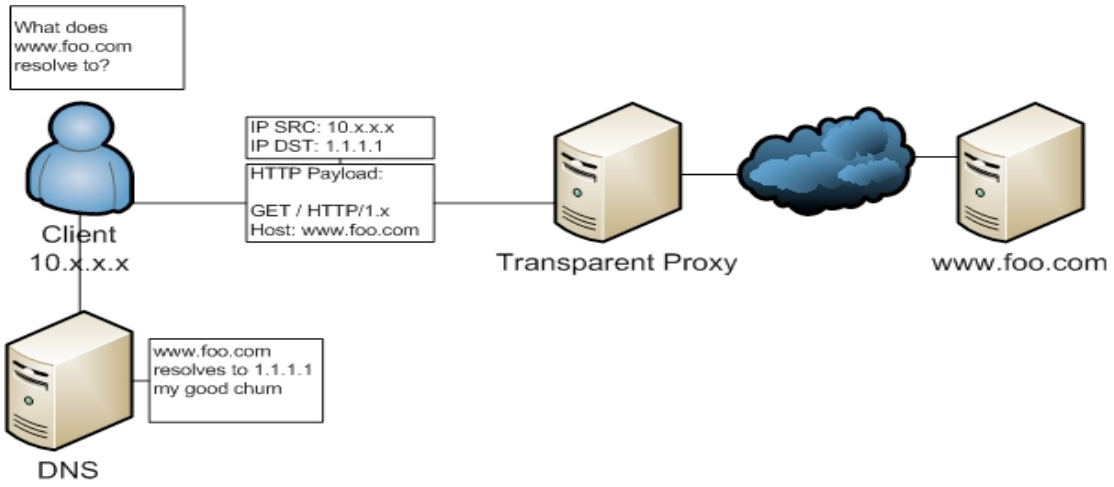
How traffic gets to transparent proxies

- Technologies such as WCCP/GRE/IPTables/IPFW are often used to force/redirect traffic to the transparent proxy
  - The user is unaware this is going on
- Proxy is typically on a dedicated machine, sometimes deployed on the gateway/router itself
- Often involves rewriting the packet's destination to the proxies IP address and port (NAT)
- Some implementations merely sniff the wire and may not terminate to a service
- If the proxy is listening on all addresses then rewriting shouldn't be required, although it is unknown how common this approach is

Common transparent proxy architectures

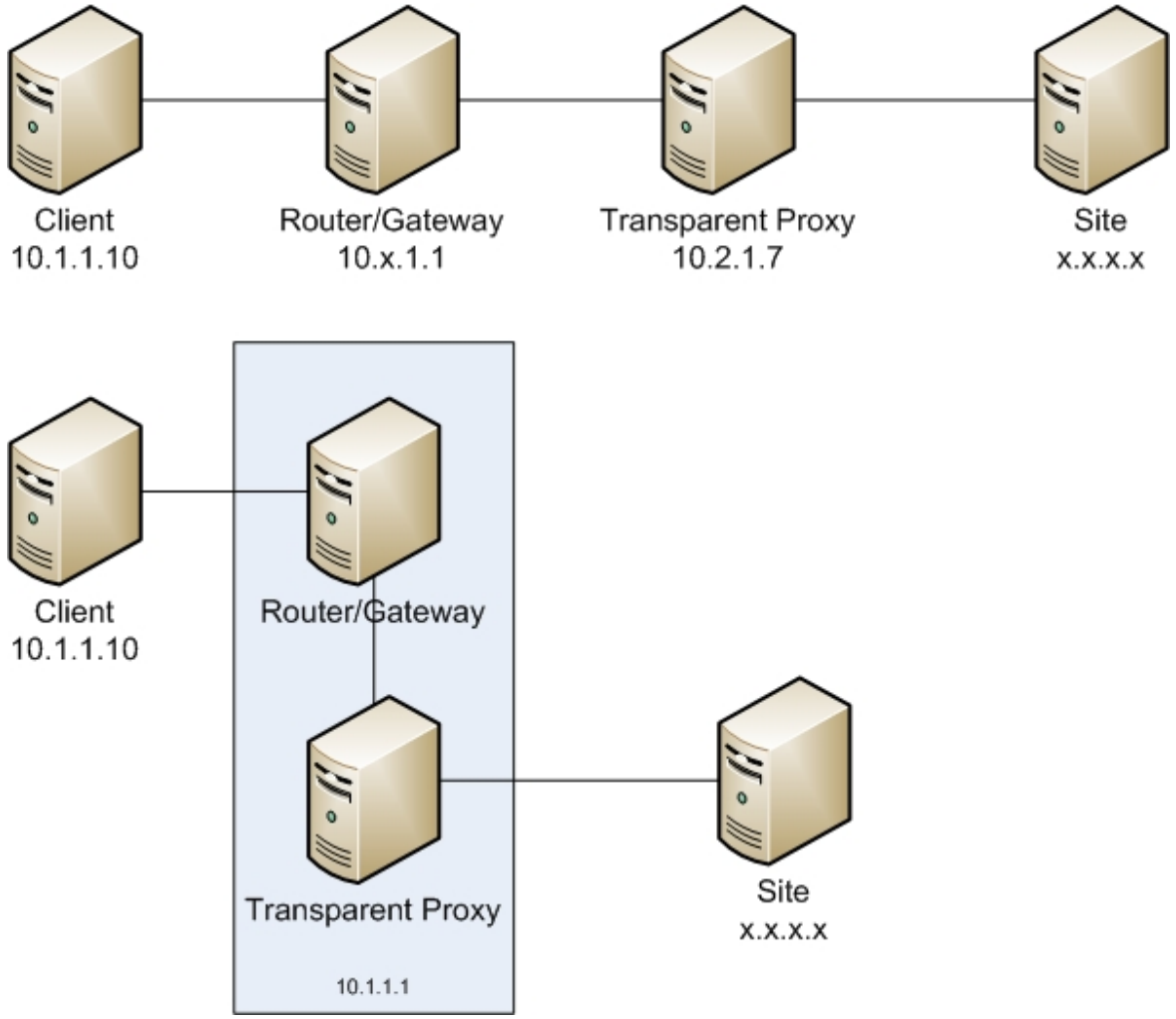
Approach A: Use the destination IP from the client

- Proxy server determines destination based on original destination-IP address of client request. In this configuration the transparent proxy routes requests much like a standard router by basing its routing decisions off of the network layer (layer 3).



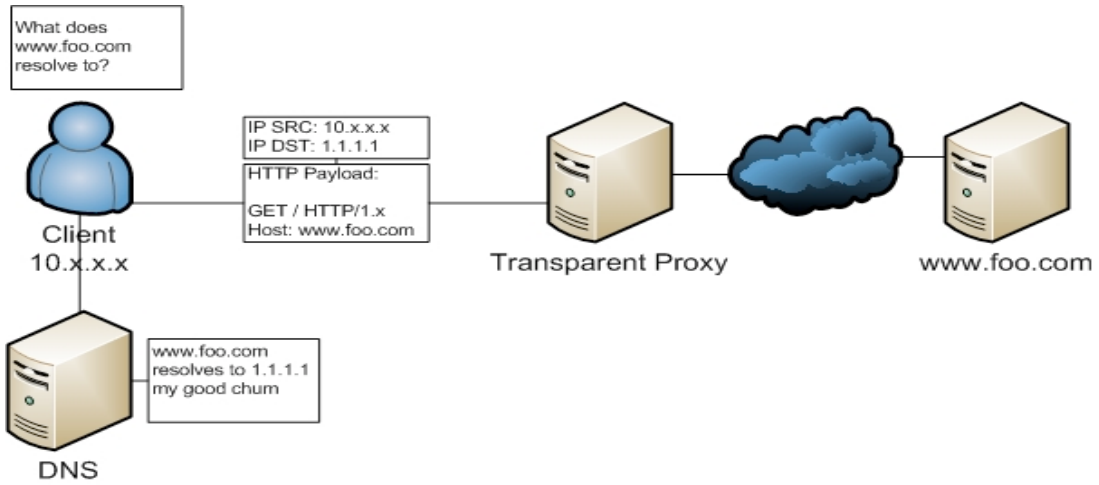
- Architecture is defined in RFC 1919 (*Classical versus Transparent IP Proxies*) which is marked 'Informational' and is not a standard.
- Only can be used in certain network architectures.

When 'Approach A' cannot be implemented (NAT)



Approach B: Inspect application layer data

- Proxy server determines destination based on the HTTP Payload from the client request. In this configuration the transparent proxy is determining IP destinations based on the application protocol (layer 7) instead of IP (layer 3).



- Architecture not defined in any standard including RFC 2616 (HTTP 1.1).
- Due to the socket capabilities of browser plug-ins (flash/etc) this second architecture can be exploited by an attacker to gain access to any destination accessible by the proxy.

Brief intro into the Same Origin Policy (SOP)

- **A policy which permits scripts running on pages originating from the same site to access each others methods and properties without restrictions**
  - Site A can access Site A's other content
  - Site A generally can't access Site B's content due to being on a different site/origin
- The same origin policy is designed to restrict a site's access to itself
- Without the SOP Site A would be able to make requests to Site B and see the full response and Cookie data
- Technologies such as Silverlight and Flash have their own variants of the SOP enforced outside of the browser
  - Flash: Crossdomain.xml files and Socket policy files

Flash and Sockets

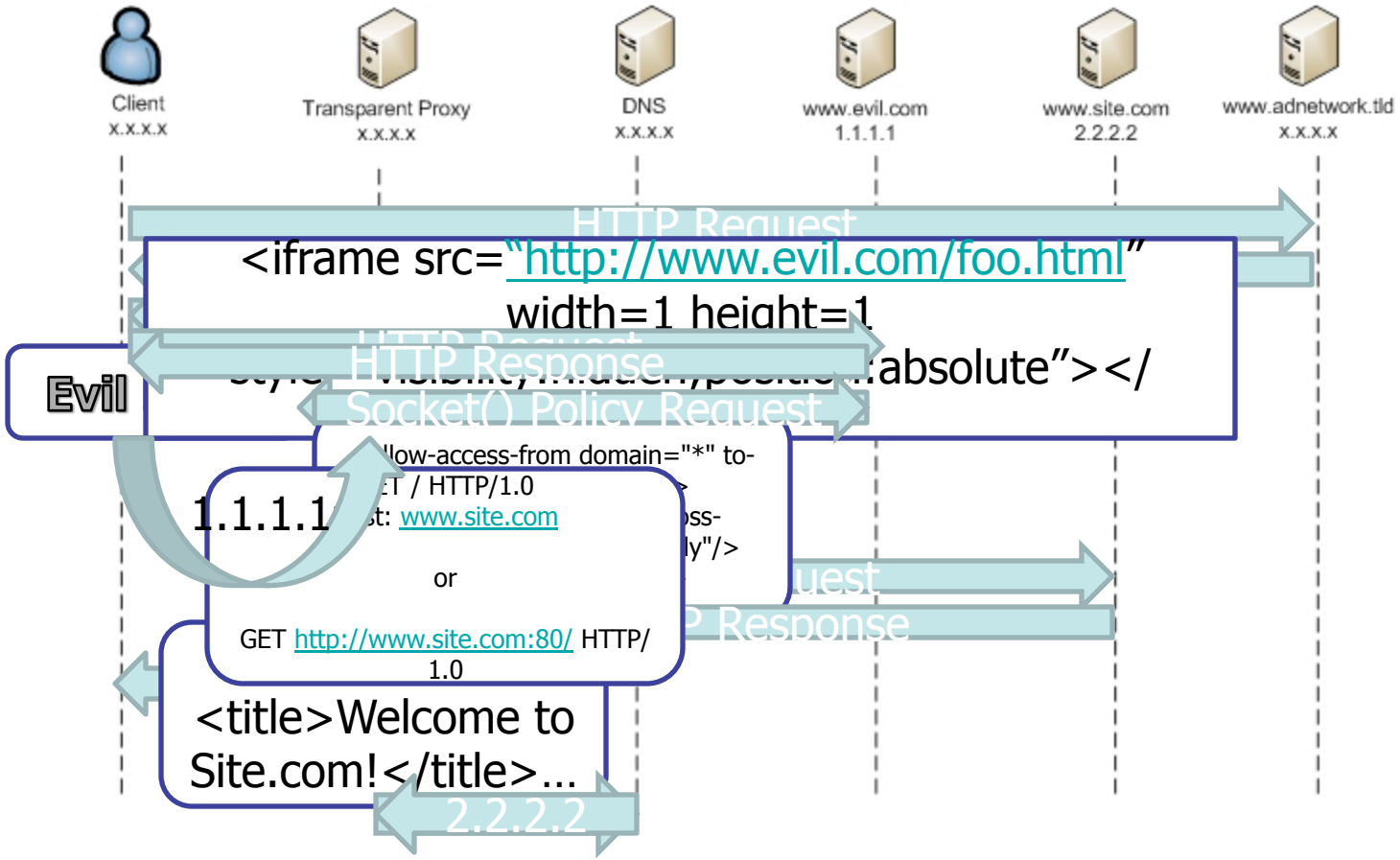
- Starting in Flash 9 Flash implemented socket policy files (*socket master policy files)* even for requests back to the same site hosting the flash
- Usually hosted on a socket policy server and not served up via HTTP
- Flash checks TCP port number 843 by default for this policy
  - Can specify a custom location with Flash's loadPolicyFile() call
- Same general format as crossdomain.xml
  - <cross-domain-policy> <allow-access-from domain="my.com" secure="true" to-ports="3050"/> </cross-domain-policy>
  - More info at http://www.adobe.com/devnet/flashplayer/articles/fplayer9_security_04.html

Recap

- At this stage we've reviewed
  - Proxy types and how their destination logic
  - Same Origin Policy basics
  - Flash and its socket policy files

- Now, onto the badness!

The Abuse Case



Impact (Technical version)

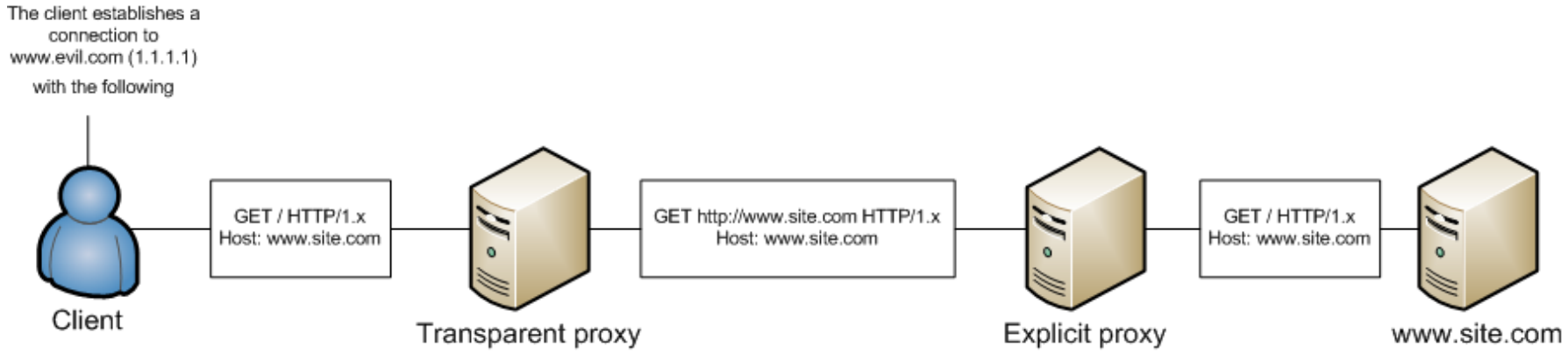
- Attacker can send HTTP Requests to any host, and obtain full HTTP responses
  - The proxy is actually making the requests, flash is used to facilitate this
- Full TCP connection support in some cases (due to CONNECT!)
  - Some intercepting proxies support explicit proxy evasion as a feature to prevent people in an organization from using an external proxy, effectively falsifying the explicit proxy connection and tricking the client.
  - ISP's unlikely to implement proxy avoidance (china maybe?)
  - Depends on port restrictions for CONNECT method on the proxy
- Limitations
  - Cookies and HTTP auth will not be obtainable because the SOP context is under www.evil.com
  - Auth can be negotiated manually (brute forcing)
    - NTLM/Basic/HTTP Based

Impact (Marketing version)

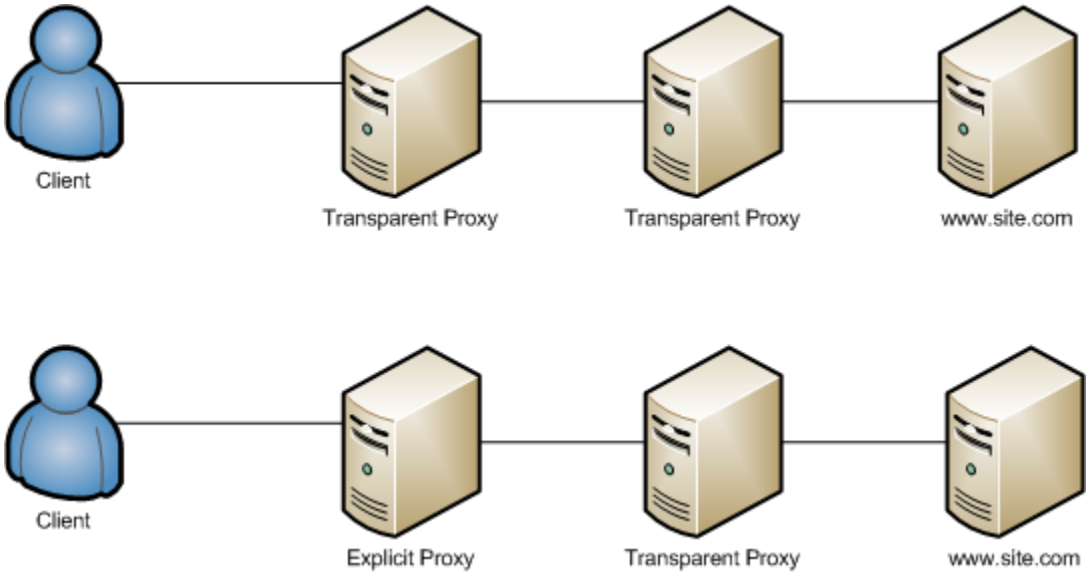
- Turn browsers into temporary botnet members
  - Only for the length of time the malicious flash is loaded
  - Perform brute forcing
  - DDOS Flooding
    - HTTP based DOS
    - TCP connection based DOS
  - Hard to track due to the lack of installed malware
    - Limited lifetime
- Intranet TCP port scanning@#!
  - When the transparent proxy is on the local network, or LocalHost
    - Otherwise allowed to make arbitrary TCP connections to internet
- Allow an attacker inside access to your network evading any NAT/ firewalling in place depending on the location of the proxy.
  - Launch other attacks against local machines from the proxy
  - Depends on the ACLS implemented on the proxy

What about proxies that are chained?

- Depending on the network architecture/proxy combination you can still be vulnerable, even if your transparent proxy is basing its destination IP on the client DST IP
- If there are two chained transparent proxies, and one bases the destination on the HTTP payload, then you're vulnerable
- Transparent proxies utilizing explicit proxies always vulnerable



Non vulnerable chained proxy configurations



Demo

<Start the video!>

Manual Reproduction

- To identify if your environment is vulnerable you can perform the following manual steps.

1. Perform a DNS lookup against a test website name
2. Telnet to that website's IP on port 80 ($ telnet <host> 80)
3. Paste the following request as the payload

```
GET / HTTP/1.0
Host: <put a different website name here>
            and/or
GET http://<differenthostname>:80/ HTTP/1.0
```

4. Hit enter twice

- It is important to specify a different website name in the 'Host' header/URI Line. If you receive content from the host specified in the host header then you're affected.

Which kinds of products can be abused by this?

- Classic Caching/Transparent proxies
  - Squid
  - Bluecoat (default configuration)
  - Netcache
  - Qbik Wingate

- Security Gateways and parental control software
  - Smoothwall, SchoolGuardian, and NetworkGuardian
  - DansGuardian (web filtering gateway)
  - Bloxx (web filtering gateway)
  - Mac OS X Parental Control Software (Pre Snow Leopard)
  - Many More

Further complications and related observations

- Flash's socket policy binds to an IP
  - Virtual hosting always going to be an issue when multiple sites share the same IP. Amit Klein wrote about these sorts of issues in multiple papers [7].
  - Cloud computing environments may share 1 IP allowing for inter cloud host abuse

- Unnamed vendors implementing web filtering and/or caching tightly incorporated 'Approach B'.
  - Made modifying the product very difficult
  - In some cases adding 'Approach A' support would introduce other issues

How I stumbled on this abuse case

- While writing a proxy scanner for work I discovered a 'bug' in my code.
  - Tool had an XML list of 'IP's to connect to' (to configure as a proxy)
  - Tool had a list of target destinations to try to connect to through the proxy (e.g 10.x.x.x, 192.168.x.x, etc…)
- An external IP (for debugging purposes) was accidentally left in my 'IP to connect to' file. When I ran the tool it flagged that it was able to access an local intranet site from my remote machine :/
  - Surprising to say the least
  - Come to find out **this bug** wasn't a bug (Wireshark/tcpdump confirmed!)
- Surely this is a known issue!?

Discovery and Coordination

- 2007 Discovery
- Early 2008 in depth research begins
- September 2008 engaged Amit Klein during Bluehat
- October 2008 Began vendor notifications
- November 2008 handed disclosure and notifications over to CERT(R) Coordination Center
- December 2008 Contacted Dan Kaminsky and began joint discussions with Amit Klein, and Adrian Chadd
- February 2009 CERT Publishes advisory
- March 9th 2009 Whitepaper published
- March 10th 2009 Buzzword contest held

Buzzword contest winners

- The day after the paper was released I held a contest to name that talk/buzzword!
- Why? Because every security flaw has to have a jazzy buzzword associated with it otherwise you're not a real application security professional</sarcasm>
- Actually I'm pretty damned sick of security industry buzzwords so included this slide to poke fun at them
- Buzzword winners: ProxyJacking, PITM (Proxy in the middle attack)
- Talk name winner: 'Down with O.P.P. - other people's proxy'
- More amusing submissions @ http://www.cgisecurity.com/2009/03/proxy-attack-stupid-buzzword-contest-.html

Fundamental problem

- Transparent proxies fundamentally alter the security assumptions
  - No internet standards that define transparent/ intercepting proxies
    - RFC 2616 touches on it
- Proxy can't tell the difference between browser and flash
- Flash can't tell when a transparent proxy is being utilized
- Dan Kaminsky has published related research at http://www.doxpara.com

Who *should* fix this?

- Opinions vary on who is responsible
  - **Proxy Vendors?**
    - Fair to say this only exists due to client side plugins with socket support
    - The deployment scenario is a factor (traffic redirection is the sysadmin's decision)
  - **Client Side plug-in vendors with socket support?**
    - It is fair to say they couldn't predict an intercepting proxy exhibiting this behavior, after all intercepting proxies aren't documented in a standard!
  - **The HTTP standards?**
    - Fair to say that client side socket support wasn't anticipated, and breaks the usage model
    - There is no standard that I could find outlining the various approaches, and their pro's and cons for transparent/intercepting proxying
    - Clarity on intercepting proxies would be useful
  - **Sysadmins?**
    - Network configuration plays a part

Proxy vendor fix approaches

- Emulate the IP destination, or Pass-through to it
  - Bluecoat utilizes this method [12]
  - Depends on the redirection/pass through configuration
  - Many vendors lack the access required to gather this information
    - Kernel/Driver
- Verify that the host in the URI resolves to at least one destination IP provided by the client
  - DansGuardian utilizes this method [10]
  - Approach causes problems with round robin DNS
    - Sites may become unreachable
    - Squid team mentioned this in our discussions as well
- Limit ports and implement ACLs
  - Always a good thing to do
- Research new methods
  - New HTTP header to communicate the client DST IP?
    - X-RemoteIP?
    - X-ProxyClientDestinationIP?
      - Came up while chatting with Adrian Chadd from SQUID
      - Specific to chained proxy environments
- Ultimately depends on the network/proxy deployment configuration
- Proxy vendors are currently in the best position to outline the advantages and risks to each approach

Client-side technology fix approaches

- Disable socket functionality (yeah, right!)
- Restrict the ports sockets connections can access
  - Silverlight implements this restriction on ports 4502-4534 [6]. Flash has no such restrictions.
  - Can still access web servers/TCP services on high ports .
    - Albeit much lower chance
- Vendors need to investigate and enhance their restriction models

How can I protect my environment? (Sysadmins)

- No matter what your configuration is be sure to implement ACLS on your proxy and disallow access to sensitive hosts
- Implement port restrictions on your proxy when possible
  - Can help to prevent port scanning and connecting to other services through the proxy
- Check with your product vendor on system hardening approaches
  - Bluecoat has trust-destination-ip ignoring the host header [9]
  - Smoothwall is investigating a fix [11]
- Evaluate your transparent proxies usage, and evaluate your environment's susceptibility

How can I protect myself? (Users)

Security industry *answer* (impractical answer)

- Disable Flash/Java/Silverlight and run client side plugins like NoScript
- Use telnet and Stunnel/Openssl for web surfing

Practical answer

- Ensure your software is patched up to date
  - Parental control software
  - Web security software
  - Web browser
  - Software such as Flash/Silverlight/Java/Etc

Conclusions

- Client side Socket functionality facilitates this abuse case. Client side sockets aren't going away anytime soon and are actually expanding (WebSocket standard in HTML5)
- If every client side technology vendor with Socket() support removed sockets, or found a *better way* of implementing them this abuse case would go away. Until then we have to work around it, and be aware of it.
- A venue doesn't exist for documenting and standardizing the differences and risks between network designs, or proxy use cases (not in an RFC)
- A one size fits all *fix* simply isn't possible.
  - Not every combination of network and proxy deployment scenario's have a 'fix' available
    - Certain NAT, and proxy chain configurations
- Bottom line is some products can be abused and others can't due to the
  - Proxy location on the network/ network design
  - Proxy architecture utilized in the product. Some products only have one mode.
- More in depth information can be found in my whitepaper [1]

About me

- Play the part of an application security engineer at PayPal where I work on security testing approaches, and SDLC strategies
- Co Founder of The Web Application Security Consortium (http://www.webappsec.org/)
- WASC Threat Classification Project leader (my second job)
- (http://www.webappsec.org/projects/threat/)
- Founder and Moderator of The Web Security Mailing List (http://www.webappsec.org/lists/websecurity/)

Questions?

References

- Socket Capable Browser Plugins Result In Transparent Proxy Abuse by Robert Auger
  [1] http://www.thesecuritypractice.com/the_security_practice/2009/03/socket-capable-browser-plugins-result-in-transparent-proxy-abuse.html

- Intercepting proxy servers may incorrectly rely on HTTP headers to make connections
  [2] http://www.kb.cert.org/vuls/id/435052

- Staring Into The Abyss: Revisiting Browser v. Middleware Attacks In The Era of Deep Packet Inspection
  [3] http://www.doxpara.com/abyss

- "WebSockets" considerations and discussions about this issue
  [4] http://www.ietf.org/mail-archive/web/hybi/current/msg00031.html
  [5] http://www.ietf.org/mail-archive/web/hybi/current/msg00032.html

References (Cont)

- Why does Silverlight have a restricted port range for Sockets?
  [6] http://blogs.msdn.com/ncl/archive/2009/06/23/why-does-silverlight-have-a-restricted-port-range-for-sockets.aspx

- Write-up by Amit Klein: "Forging HTTP request headers with Flash"
  [7] http://www.webappsec.org/lists/websecurity/archive/2006-07/msg00069.html

- CVE
  [8] CVE-2009-0801, CVE-2009-0802, CVE-2009-0803, CVE-2009-0804

- ProxySG in transparent deployments intercepting HTTP/HTTPS traffic
  [9] https://bto.bluecoat.com/support/securityadvisories/ProxySG_in_transparent_deployments

- DansGaurdian Changelog
  [10] http://dansguardian.org/?page=history

- SmoothWall Information for VU#435052
  [11] http://www.kb.cert.org/vuls/id/MAPG-7M6SM7