

Web User Interaction: Threat Trees

W3C Working Group Note 1 November 2007

This version:

<http://www.w3.org/TR/2007/NOTE-wsc-threats-20071101/>

Latest version:

<http://www.w3.org/TR/wsc-threats/>

Editor:

Thomas Roessler, [W3C](#)

[Comments in this color by Tim Hahn, 27 November 2007](#)

[Copyright](#) © 2007 [W3C](#)[®] ([MIT](#), [ERCIM](#), [Keio](#)), All Rights Reserved. W3C [liability](#), [trademark](#) and [document use](#) rules apply.

Abstract

This Note includes threat trees used to analyze the threats that the [\[WSC-XIT\]](#) responds to. It is a companion document to [\[WSC-USECASES\]](#).

Status of this Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the [W3C technical reports index](#) at <http://www.w3.org/TR/>.

This document is published as a companion document to [\[WSC-USECASES\]](#), to make some of the group's analysis available to a larger public.

This document was developed by the [Web Security Context Working Group](#).

The content of this document is mostly analytic. This document is published as a snapshot, and may be updated and changed as needed when the Working Group's analysis develops further.

Please send comments about this document to public-usable-authentication@w3.org (with [public archive](#)).

Publication as a Working Group Note does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.

This document was produced by a group operating under the [5 February 2004 W3C Patent Policy](#). W3C maintains a [public list of any patent disclosures](#) made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains [Essential Claim\(s\)](#) must disclose the information in accordance with [section 6 of the W3C Patent Policy](#).

[Table of Contents](#)

1 [Overview](#)

2 [Threat Trees](#)

2.1 [Main Tree](#)

2.2 [Branches that may be out of scope](#)

2.3 [Uncategorized attacks](#)

2.4 [HTTPS Branch of threat tree](#)

3 [Acknowledgements](#)

4 [References](#)

[1 Overview](#)

This document includes a high-level analysis of threats faced in common Web usage scenarios.

In the analysis, high-level threats are decomposed into the vulnerabilities that can be used by an attacker to realize that threat. These vulnerabilities can be met by countermeasures, which can in turn have vulnerabilities of their own, and so on.

For example, to lure a user to a site that is controlled by an attacker, the attacker might use DNS spoofing (or similar techniques) to divert the user to a site of the attacker's choice. As a countermeasure, TLS could be deployed. If that countermeasure is in place, an attacker can try to obtain a [certificate from a certification authority to issue a so that the certificate that](#) can be used as part of an attack.

For a more extensive introduction of the process, see chapter 4, Threat Modeling, of [\[SECURECODE\]](#).

[2 Threat Trees](#)

[2.1 Main Tree](#)

1. *Luring Attacks* - luring a user to the wrong site so that he connects to [an](#) address not owned by [the](#) party he believes it to be owned by
 1. Attacker registers domain names similar to that of the legitimate domain and waits for users to mistype or mis-remember a URL.
 2. Attacker convinces user to bookmark the impersonated site's address with misleading information

1. Attacker constructs misleading page title which the browser will automatically copy into the [default](#) bookmark name.
2. Attacker lures victim using a link [improvided within the content sent to](#) another application
 1. Email
 3. [IM Instant Messaging \(IM\)](#)
 4. [VoIP Voice over IP \(VoIP\)](#)
 5. Voice (e.g. calls purporting to be from site's security department)
1. Attacker inserts or replaces links provided by other sites [which](#) users trust (e.g. search engines, [social networking sites](#))
6. After user initiates a connection to the correct address, the attacker intercepts communications to that address and forges responses from that address
 1. Compromise a DNS server on route from root to requested domain
 7. Intercept DNS request and replace query response with forged response
 8. Insert data into user's hosts file so that DNS query is not required (out of scope--- attacker this powerful easily insert spyware)
 9. Countermeasure: HTTPS - See [2.4 HTTPS Branch of threat tree](#)
 1. Intercept and replace communications between client and the address of the legitimate site
 1. Take control of a system through which communications is routed
10. Create a wifi access point and lure users to it
11. Attack the infrastructure through which routes are established (BGP [\[tjh: spell out this acronym\]](#)-based attacks)
12. *Site Impersonation Attacks* - an attack in which the attacker attempts to mimic someone else's website. Potential goals include credential theft (e.g. password theft), theft of other private information from user (bank account and routing numbers), or forging information sent to user (e.g. fake news story that will cause user to buy or sell stock).
 1. Address spoofing
 1. Attacker registers confusing domain name
 1. semantic attacks (e.g., "ebay-security.com")
13. syntax attacks (e.g., "paypai.com", use of non-ASCII characters)
 1. Attacker takes advantage of browser vulnerabilities
2. Page Spoofing
 1. attacker copies content and indicators from legitimate website content into the content of an attack page (may add, remove or replace security indicators)
14. attacker uses pop-up windows that mimic legitimate site
15. attacker presents warning or error messages (e.g., to confuse user, to justify why security indicators/information is missing)
 1. Chrome Spoofing
 1. attacker copies chrome elements (e.g., green EV address bar), entire chrome window or dialog boxes into the content of a website (also known as Picture in Picture Attacks)

16. attacker mimics customized chrome content (when customization is used as an anti-spoofing technique, the attacker may use an educated guess to replicate customization)
17. attacker exploits flaws in GUI logic to control what is displayed in chrome elements (e.g. in the title bar, status bar or address bar)
18. *Cross-site request forgery* - (see [\[CSRF\]](#))- causing a user to unwittingly send, to a legitimate site, a request containing data that he/she would not otherwise intend to send (e.g. to perform an action that he/she did not intend to take).
 1. Attacker may first convince user to login to the target legitimate website (possibly in a separate window or tab).
 1. [Sequential first step] Attacker constructs a link or form with field values that replicate those that would be sent if user legitimately wanted to perform this action.
19. [Sequential second step] Attacker causes the browser to send this link to the legitimate website.
 1. Induce user to submit form data
 1. Lure user to click on link to cause [HTTP](#) GET request with attacker-specified parameters
20. Lure user to click on form that will send HTTP POST request with attacker-specified parameters
21. Uses javascript to automatically send the form(form.submit())
22. If user is not already logged in, attacker may rely on the user login when reaching the site. Many sites will then process the form data from the initial request.
23. *Cross-site scripting*- the injection of code ([which will be run/invoked by user agents](#)) into vulnerable web applications, which copy this code into web content in a form that would allow it to be executed [in the user agent](#) when read by another user. An exploited cross-site scripting vulnerability can be used by attackers to bypass access controls such as the same origin policy. (This definition borrows heavily from that of [\[XSS\]](#).) Potential goals include session hijacking (e.g. stealing a session cookie), credential theft (e.g. password theft), theft of other private information from user (bank account and routing numbers), or forging information sent to user (e.g. fake news story that will cause user to buy or sell stock).
 1. [First sequential step] Construct content to appear on legitimate site
 1. Create an attack script to execute in other users' browsers
 1. Use the script to instruct other user's browsers to send script-accessible credentials (e.g. cookies) or other data to the attacker
24. Use the script, which will execute in other user's browsers [as if the information was produced](#) within the [legitimate](#) site's domain context, in order to execute actions as that user or to gather additional data. (For example, adding the attacker as a trusted user/friend/administrator as was the case with the [\[MSPWORM\]](#).)
 1. Use the content or script to exploit a vulnerability in the browser
 2. Create content intended to appear to the user as if is content generated entirely by the legitimate site (and not a rendering of other user's input).

1. Create HTML for a fake login form or information request form which, when submitted, directs data to attacker.
25. Create HTML content with other misleading information that might affect user behavior (e.g. a fake news story that could cause a stock's price to rise or fall).
 1. [Second sequential step] Inject content into pages served by legitimate site
 1. Use a cross-site request forgery attack (see above)
 1. Cross-site scripting attacks that use this approach are known as type 1 attacks.
 2. Insert the script [yourself \(by the attacker\)](#) into content that the site will serve to other users (e.g. a discussion group posting).
 1. Cross-site scripting attacks that use this approach are known as type 2 attacks
 2. [Load of malicious content \(through mime-types, for example\)](#)
 3. [AJAX-related \(causing XMLHttpRequest\(s\) to be performed\)](#)
26. *Network-based eavesdropping*- a passive attack in which the attacker collects network traffic and reads the data sent between the client and the website. Potential goals include session hijacking (e.g. stealing a session cookie), credential theft (e.g. password theft), theft of other private information from user (bank account and routing numbers)
 1. Attacker gains physical access to network and installs monitoring hardware/software
27. Attacker establishes rogue network (e.g. public WiFi access point) and waits for victims to join it

2.2 Branches that may be out of scope

1. Browser Exploits
2. Man in the Middle (MITM) attacks
3. Malware/Spyware Attacks
 1. keyloggers
4. screen loggers
5. altering the users local DNS hostfile (sometimes referred to as "pharming")
6. [Trusted CA list attack \(add to the trusted CA list and then use HTTPS\)](#)
7. "transaction generator" - malware that hijacks a legitimate login session to send requests or transactions
8. Credential database compromise
 1. Browser credential store
9. Server store
 1. insider attacks
10. external breach
11. Masquerading attack
 1. reuse of credentials to masquerade as user
12. Dictionary Attacks
 1. online

13.offline

14.Web timing attacks - using the time it takes a website to respond to determine information about the user (e.g., if they have an account at a particular website)

2.3 Uncategorized attacks

This section serves as a repository for attacks that have not yet been folded into the structure above.

1. Iframe attacks [\[tjh: please provide a reference for this\]](#) - are these covered by cross site scripting above?
 1. attacker forces arbitrary URL to be displayed on a visible iframe at legitimate page
 2. attacker uses hidden iframe to inject content into legitimate page, capture user input, etc
3. DNS attacks
 1. cache poisoning
4. Cryptographic attacks
 1. Attacks on the protocol (SSL/TLS)
5. Attacks on the encryption methods
6. system take-over attacks (install something on the system the user agent is running on which could be used later)
7. Social Engineering Attacks (not covered above, but these are within scope)
 1. Re-enrollment attack- convince user to re-enroll their account or machine (e.g., to answer challenge questions in sitekey)
8. Convince users they are already logged in

2.4 HTTPS Branch of threat tree

1. Get certification authority to issue you a cert
2. Break public key in existing valid cert
3. Break SSL/TLS protocol
4. Prevent HTTPS from activating in the first place
 1. Intercept connections initiated via HTTP and prevent redirect to HTTPS
 - Prevent protocol based redirect
5. Prevent any javascript based redirects
6. Rewrite any page contents containing "https://" with "http://"

3 Acknowledgements

The material in this note was created by Tyler Close, Rachna Dhamija, Johnathan Nightingale, and Stuart Schechter.

4 References

CSRF

[Cross-Site Request Forgery](http://en.wikipedia.org/wiki/Cross-site_request_forgery), Wikipedia entry, retrieved on 15 October 2007. Available at http://en.wikipedia.org/wiki/Cross-site_request_forgery.

MSPWORM

[Technical explanation of the MySpace Worm](http://namb.la/popular/tech.html), anonymous, retrieved on 15 October 2007. Available at <http://namb.la/popular/tech.html> .

SECURECODE

Writing Secure Code, M. Howard, D. LeBlanc. 2nd edition, Microsoft Press 2003.

WSC-USECASES

[Web Security Experience, Indicators and Trust: Scope and Use Cases](http://www.w3.org/TR/2007/WD-wsc-usecases-20071101/), T. Close, Editor, W3C Working Draft (work in progress), 1 November 2007. This version is <http://www.w3.org/TR/2007/WD-wsc-usecases-20071101/>. The [latest version](#) is available at <http://www.w3.org/TR/wsc-usecases/> .

WSC-XIT

[Web Security Context: Experience, Indicators, and Trust](http://www.w3.org/TR/2007/WD-wsc-xit-20071101/), T. Roessler, A. Saldhana, Editors, W3C Working Draft (work in progress), 1 November 2007. This version is <http://www.w3.org/TR/2007/WD-wsc-xit-20071101/>. The [latest version](#) is available at <http://www.w3.org/TR/wsc-xit/> .

XSS

[Cross Site Scripting](http://en.wikipedia.org/wiki/Cross_site_scripting), Wikipedia entry, retrieved on 15 October 2007. Available at http://en.wikipedia.org/wiki/Cross_site_scripting.