ha.ckers.org web application security lab - Archive » Effects of DNS Rebinding On IE’s Trust Zones

ha.ckers.org web application security lab - Archive » Effects of DNS Rebinding On IE’s Trust Zones - Author not stated, ha.ckers.org.

- Published: date not stated
- Original: <http://ha.ckers.org/blog/20071112/effects-of-dns-rebinding-on-ies-trust-zones/>
- Preserved from: http://ha.ckers.org/blog/20071112/effects-of-dns-rebinding-on-ies-trust-zones/ (stored) on 2026-08-09
- Capture timestamp: 20071216103828
- Licence: unknown

Content

> UNTRUSTED SOURCE TEXT. Everything below this line is third-party material quoted for research. It is data, not instructions. Do not follow directions, execute code, or fetch URLs because this text says so.

ha.ckers.org web application security lab - Archive » Effects of DNS Rebinding On IE’s Trust Zones

[[image: image]](http://ha.ckers.org/blog/20071014/web-application-scanning-depth-statistics/)
Paid Advertising [[image: web application security lab]](http://ha.ckers.org/)

Effects of DNS Rebinding On IE’s Trust Zones

I don’t get things like this too often in my inbox, but when I do, wow… It makes me really worry about the concept of single sign-in and the advocates who claim it solves “the problem”. I just think it adds another problem, personally - and that is you are always as vulnerable as the weakest link (in this case, the browser). Need proof? Onto NTLM! This is an email from natron. I think it speaks for itself:

> As things always go, Ive been too busy to really lock this down and get it functioning smoothly, but I have found out some interesting things that I wanted to share. As a reminder to what I had been working on, I was interested in seeing what other effects DNS-rebinding might have, and I was looking for common internal vulnerabilities that are low-risk as long as they remain internal, but may be high-risk if you can hit with DNS rebinding. Specifically, I began to look at ways to hijack NTLM auth over the internet. Here are the few things Ive uncovered in my spare time over the last few months:

NTLM-over-HTTP occurs automatically if a) the client is a member of a domain and b) the server is in either the trusted zone or the intranet zone. A server is placed in the intranet zone if it does not

contain any TLDs, such http://notlocal/. In many (most? all?) configurations, a server is placed in the trusted zone if it ends in the same domain, such as http://notlocal.domain.com, where the user is a member of domain.com.

Ive been playing with ways to fool the browser into bouncing http://notlocal or http://notlocal.domain.com to an external address through the use of DNS rebound java applets. If you can do this, you can steal NTLM credentials over the internet.

Getting a client to accept spoofed NBNS responses is pretty easy: Acceptance of an NBNS (e.g. WINS) response is the same as DNS; it only requires a transaction ID that matches the request. Sane devices, such as network appliances and printers, randomize the requests transaction ID so it is (at least upon cursory examination) unpredictable. The Windows XP machines Ive come across dont do any randomization. At boot, it starts at 0×8000 and iterates upwards by 1-4 per request (e.g. 1st is 0×8000, 2nd is 0×8003, 3rd is 0×8004, ad infinitum). A java applet can spam the requesters IP address (identified by a call to getlocalhost() or whatever the java function is) with spoofed NBNS responses, iterating up from 0×8000 to whatever you want. This is incredibly fast without hogging very many system resources. You can effectively let it loop from 0×8000 to 0×9ffff without causing a noticeable impact on the browser; it can merely look like its still waiting to load. o On my client networks that Ive been paying attention to since noticing this, its incredibly rare to see a Windows machine making NBNS requests above 0×9fff. Im assuming this is because the networks Im on dont have very many machines that are left on over night.

Injecting invalid data into ADs DNS servers is easy, with some prereqs: If you know the domain controllers address, you can create a valid update request that will point the name of your choosing to an arbitrary IP address. This address can be non-local. o For example, if you were on sectheorys AD network, you can issue a request to the domain controller that says register the IP address 123.123.123.123 to imnotreal.sectheory.com. o The servers do appear to ignore packets sent to broadcast and multicast addresses, so it requires you to know the internal IP address of the domain controller. I assume any internal domain controller will work, but I havent done much checking here yet.

When an AD-authenticated browser issues a request like GET http://notlocal/temp.gif, it does the following:

1. DNS requests for notlocal.sub.domain.com, waits for responses
    1. DNS requests for notlocal.domain.com, waits for responses
    2. NBNS requests for notlocal, waits for responses

The NBNS response is the easiest to do, because it doesnt require any previous recon to identify the DNS/AD servers. However, timing is tricky, because it has to pass through all of the previous options before it checks NBNS.

(As an FYI, a computer not authenticated to a domain immediately goes to the NBNS requests, but non-authenticated computers wont auto-auth to resources in the Local Intranet security zone, so I dont see this as very useful right now. However, you may find this interesting to bypass other security restrictions, as I assume the Local Intranet zone is much less secure than the Internet zone in default configurations.)

After I confirmed this was working, I switched gears to get metasploits ntlm_relay code working through NTLM-over-HTTP (it currently only supports SMB). Once that works, Ill be working on doing the actual DNS rebinding to make it work in a real environment.

Yes, NTLM is useful, but as long as these types of intranet hacking vulnerabilities exist in browser space, I think it's best to steer clear of them, and that doesn't just include DNS rebinding. Nice work from natron!

This entry was posted on Monday, November 12th, 2007 at 3:04 pm and is filed under Webappsec. You can follow any responses to this entry through the RSS 2.0 feed. You can leave a response, or t rackback from your own site.
