Interactive DNS rebinding

Yunxing Dai, Ryan Resig
yunxing@upthere.com, rresig@umich.edu
About myself

• Just graduated from University of Michigan
• Joined upthere as a backend engineer
王慎辉

to 165402403, wanglu, cq, me, 640708360, cstsintern, 29561390,

http://disenowebjoomla.com/hojofgu/uoenrkkfs.jiskvbepdxvk
How secure is it?

• Given a malicious link, how secure is your browser?
• Incognito mode
• sandbox
• no input of sensitive data
• no executables files
Goal

• Allows the attacker to make the victim’s browser as a proxy, thus gain access to an internal webpage
Level-0
Level-0
Level-0
Level-0

```javascript
<script>
  // Send HTTP Request
  $.get('127.0.0.1', function(data) {
    // Send data to the attacker's server
  });
</script>
```
Level-0

<script>
    // Send HTTP Request
    $.get("127.0.0.1"), function(data) {
        // Send data to the attacker's server
    }
</script>

Firewall

Victim Web Server

Tacker Server

yunxing@upthere.com
Level-0

<script>
   // Send HTTP Request
   $.get("127.0.0.1");
   function(data) {
      // Send data to the attacker's server
   }
</script>
Level-0

Code snippet:
```
<script>
  // Send HTTP Request
  $.get("192.0.0.1:", function(data) {
    // Send data to the attacker's server
  });
</script>
```

Diagram showing:
- Browser
- Victim Web Server
- Firewall
- Web
- Attacker Web Server

yunxing@upthere.com
Level-0

[Diagram showing a network setup with layers labeled Victim, HTTP Request, Victim Web Server, Web, Firewall, and Attacker Web Server.]
Level-0
Level-0

[yunxing@upthere.com]
Level-0

Diagram:
- Victim Browser
- Victim Web Server
- Firewall
- HTTP Response
- Attacker Web Server

Contact:
yunxing@upthere.com

Thursday, August 22, 13
Level-0
Same Origin Policy

XMLHttpRequest cannot load http://127.0.0.1/.
Origin http://n1.takenoteswith.us is not allowed
by Access-Control-Allow-Origin.
DNS Rebinding

- Attack the system in the IP layer.
- Time varying
Level-1 Time-Varying

attacker.com = ?

Browser

Query: attacker.com

Victim Web Server

Apache

Web

Firewall

Attacker Domain Server

Attacker Web Server

1.2.1.2

Thursday, August 22, 13
Level-1 Time-Varying

attacker.com = ?

Victim Browser

Victim Web Server

Query: attacker.com

Firewall

Attacker Domain Server

Attacker Web Server

1.2.1.2

Thursday, August 22, 2013
Level-I Time-Varying

attacker.com = ?

Victim Browser

Firewall

Web

Query: attacker.com

Domain Server

Attacker Web Server

1.2.1.2
Level-1 Time-Varying
Level-I Time-Varying
Level-1 Time-Varying

attacker.com = ?

Victim Browser

Firewall

Web

Attacker Domain Server

1.2.1.2

Victim Web Server

Attacker Web Server

Resps: 1.2.1.2
TTL = 0

Thursday, August 22, 13
Level-1 Time-Varying

attacker.com = 1.2.1.2

Firewall

Victim Browser

Apache

Victim Web Server

Web

Attacker Domain Server

Atacker Web Server

1.2.1.2
Level-I Time-Varying

```html
<script>
// Send HTTP Request
$.get("attacker.com"), function(data) {
    // Send data to the attacker's server
}
</script>
```
Level-1 Time-Varying

// Send HTTP Request
$.get("attacker.com");
// Send data to the attacker's server

1.2.1.2
Level-1 Time-Varying
Level-I Time-Varying

attacker.com = ?

Browser

Query: attacker.com

Firewall

Web

Attacker Domain Server

Attacker Web Server

1.2.1.2

Victim Web Server

Apache

yunxing@upthere.com
Level-I Time-Varying

attacker.com = ?

Victim Browser

Query: attacker.com

Victim Web Server

Firewall

Attacker Domain Server

Attacker Web Server

1.2.1.2

yunxing@upthere.com

Thursday, August 22, 13
Level-I Time-Varying

```
attacker.com = ?
```

```
Firewall
```

```
Query:
attacker.com
```

```
Attacker
Domain
Server
```

```
Victim Web Server
```

```
Apache
```

```
Victim Browser
```

```
Web
```

```
1.2.1.2
```

Thursday, August 22, 13
Level-1 Time-Varying

attacker.com = ?

Victim Browser

Victim Web Server

Firewall

Web

Attacker Domain Server

Respond: 127.0.0.1

Attacker Web Server

1.2.1.2

yunxing@upthere.com
Level-I Time-Varying

```
attacker.com = ?
```

```
Victim Browser

Victim Web Server
```

```
Firewall
```

```
Respond: 127.0.0.1
```

```
Attacker Domain Server

1.2.1.2
Attacker Web Server
```

```
yunxing@upthere.com
```

Thursday, August 22, 13
Level-I Time-Varying

 attacker.com = ?

 Victim B

 RESPONS: 127.0.0.1

 Victim Web Server

 Apache

 Firewall

 Web

 Attacker Domain Server

 1.2.1.2

 Attacker Web Server

 yunxing@upthere.com

 Thursday, August 22, 13
Level-1 Time-Varying

attacker.com = 127.0.0.1

 Victim Browser

 Victim Web Server

Attacker Domain Server

Attacker Web Server

1.2.1.2

Thursday, August 22, 13
Level-1 Time-Varying

```html
<script>
  // Send HTTP Request
  $.get("attacker.com"), function(data) {
    // Send data to the attacker's server
  }
</script>
```
DNS Pinning

- Ignore the TTL and pin the DNS entry in the Browser’s cache after it expires
Time varying attack

- Force the browser to issue the second DNS query
Same Origin Policy?
Same Origin Policy?

• Screws API up!
Same Origin Policy - Exceptions

- JSONP
- WebSocket
Same Origin Policy - Exceptions

- JSONP
- WebSocket
Same Origin Policy - Exceptions

- Another DNS query
Same Origin Policy - Exceptions

- Which will be pinned to the browser
Same Origin Policy - Exceptions

- The Browser has small DNS cache size
Same Origin Policy - Exceptions

• Earlier entries will be evicted
Same Origin Policy - Exceptions

- And we can thus flood it
Implementation

- Flood the DNS cache with lots of cross site references
- Evict the original DNS entry
- Force the browser to do the second DNS query
Level-2 DNS table flooding

Attacker.com = ?

- Browser
- Victim Web Server
- Attacker Web Server
- Firewall
- Web

Attacker Domain Server

1.2.1.2

yunxing@upthere.com

Thursday, August 22, 13
Level-2 DNS table
flooding
Level-2 DNS table flooding

Victim Browser -> Victim Web Server -> Web

Firewall

Query: attacker.com

Attacker Domain Server

Attacker Web Server

1.2.1.2

yunxing@upthere.com
Level-2 DNS table flooding

- Victim Browser
- Victim Web Server
- Attacker Web Server
- Attacker Domain Server
- Firewall
- Web

attacker.com = ?

Respond: 1.2.1.2 TTL = 0

1.2.1.2

yunxing@upthere.com
Level-2 DNS table flooding
Level-2 DNS table flooding
Level-2 DNS table flooding

attacker.com = 1.2.1.2
Level-2 DNS table flooding

```html
<script>
    for (var i = 0; i < 1024; ++i) {
        // Flush the DNS table
        $.get("n" + i + ".attacker.com"), function() {
        }
    }

    // Send HTTP Request
    $.get("attacker.com"), function(data) {
        // Send data to the attacker's server
    }
</script>
```
Level-2 DNS table flooding

- Victim Web Server
- Attacker Domain Server
- Attacker Web Server

Attack: Send requests to the victim's server, then route them to the attacker's server.

Security Measures:
- Use a firewall to block unauthorized traffic.
- Implement DNS rate limiting to prevent flooding.
Level-2 DNS table flooding

```html
<script>
  for (var i = 0; i < 1024; ++i) {
    // Flush the DNS table
    $.get("n" + i + ".attacker.com"), function() {
    }
  }

  // Send HTTP Request
  $.get("attacker.com"), function(data) {
    // Send data to the attacker's server
  }
</script>
```
Level-2 DNS table flooding

`attacker.com = 1.2.1.2`

Diagram:
- **Query:**
  - n1.attacker.com
  - n2.attacker.com
  - ... 
  - n1024.attacker.com

- **Firewall**
- **Web**
- **Victim Web Server**
- **Attacker Domain Server**
- **Attacker Web Server**

Thursday, August 22, 13

yunxing@upthere.com
Level-2 DNS table flooding

attacker.com = 1.2.1.2

Query:
  n1.attacker.com
  n2.attacker.com
  ...
  n1024.attacker.com

Firewall

1.2.1.2

 Victim
Browser

 Victim
 Web
Server

 Attacker
Web Server

yuxing@upthere.com

Thursday, August 22, 13
Level-2 DNS table flooding

attacker.com = 1.2.1.2

Firewall

Victim Browser

Web

Apache

Victim Web Server

Response:
127.0.0.1
127.0.0.1
...
127.0.0.1

Attacker Web Server

1.2.1.2

yunxing@upthere.com

Thursday, August 22, 13
Level-2 DNS table flooding

attacker.com = 1.2.1.2

Response:
127.0.0.1
127.0.0.1
...
127.0.0.1

Attacker Domain Server

Attacker Web Server

Victim Web Server

Victim Browser

Firewall

Thursday, August 22, 13

yunxing@upthere.com
Level-2 DNS table flooding

attacker.com = 1.2.1.2

Response:
127.0.0.1
127.0.0.1
...
127.0.0.1

Web

Firewall

Attacker Domain Server

1.2.1.2

Attacker Web Server

Victim Web Server

Apache

Chrome
Level-2 DNS table flooding
Level-2 DNS table flooding

```javascript
<script>
  for (var i = 0; i < 1024; ++i) {
    // Flush the DNS table
    $.get("n" + i + "attacker.com"), function() {
    }
  }

  // Send HTTP Request
  $.get("attacker.com"), function(data) {
    // Send data to the attacker's server
  }
</script>
```
Level-2 DNS table flooding

attacker.com = ?

Victim Browser

Victim Web Server

Firewall

Web

Respons: 127.0.0.1
Attacker Domain Server

Attacker
Web Server

1.2.1.2

yunxing@upthere.com
Level-2 DNS table flooding
Level-2 DNS table flooding

attacker.com = ?

Victim

Respond: 127.0.0.1

Firewall

Web

Attacker Domain Server

Attacker Web Server

1.2.1.2

Apache

Victim Web Server

remen@upthere.com
Level-2 DNS table flooding

attacker.com = 127.0.0.1
Live Demo

• Registered the domain name dnsrebind.info

• Used it as a malicious link to access the router’s control panel in my apartment
More examples
More examples
More examples
Defenses

• “Host:” header field checking
• Giving DNS entry correct priorities
Discussion

• DNS rebinding has been existed for a long time, still hasn’t been fixed
Questions?

yunxing@upthere.com