How many failure reports does a bug have to get before Windows will fix it?

devblogs.microsoft.com/oldnewthing/20100804-00

Raymond Chen

When a program crashes or hangs, you are given the opportunity to send an error report to Microsoft. This information is collected by the Windows Error Reporting team for analysis. Occasionally, somebody will ask, "How many failures need to be recorded before the bug is fixed? A thousand? Ten thousand? Ten million?" Each team at Microsoft takes very seriously the failures that come in via Windows Error Reporting. Since failures are not uniformly distributed, and since engineering resources are not infinite, you have to dedicate your limited resources to where they will have the greatest effect. Each failure that is collected and reported is basically a vote for that failure. The more times a failure is reported, the higher up the chart it moves, and the more likely it will make the cut. In practice, 50% of the crashes are caused by 1% of the bugs. Obviously you want to focus on that top 1%. It's like asking how many votes are necessary to win *American Idol*. There is no magic number that guarantees victory; it's all relative. As my colleague Paul Betts describes it, "It's like a popularity contest, but of failure." Depending on the component, it may take only a few hundred reports to make a failure reach the upper part of the failure charts, or it may take hundreds of thousands. And if the failure has been tracked to a third-party plug-in (which is not always obvious in the crash itself and may require analysis to ferret out), then the information is passed along to the plug-in vendor. What about failures in third-party programs? Sure, send those in, too. The votes are still tallied, and if the company has signed up to access Windows Error Reporting data, they can see which failures in their programs are causing the most problems. Signing up for Windows Error Reporting is a requirement for the Windows 7 software logo program. I've also heard but cannot confirm that part of the deal is that if your failure rate reaches some threshold, your logo certification is at risk of revocation. If the vendor signed up for Windows Error Reporting but is a slacker and isn't looking at their failures, there's a chance that Microsoft will look at them. Towards the end of the Windows 7 project, the compatibility team looked at the failure data coming from beta releases of Windows 7 to identify the highest-ranked failures in third party programs. The list was then filtered to companies which participated in Windows Error Reporting, and to failures which the vendor had not spent any time investigating. I was one of a group of people asked to study those crashes and basically debug the vendor's program for them. My TechReady talk basically consisted of going through a few of these investigations. I may clean some of them up for

<u>public consumption and post them here</u>, but it's a lot of work because I'd have to write a brand new program that exhibits the bug, force the bug to trigger (often involving race conditions), then debug the resulting crash dump. I know there is a subculture of people who <u>turn off error reporting</u>, probably out of some sense of paranoia, but these people are ultimately just throwing their vote away. Since they aren't reporting the failures, their feedback doesn't make it back to the failure databases, and their vote for *fix this bug* never gets reported. By the way, the method for disabling Windows Error Reporting given in that linked article is probably not the best. You should use the settings in the Windows Error Reporting node in the Group Policy Editor.

I should've seen this coming: Over time, I've discovered that there are a some hot-button topics that derail any article that mentions them. Examples include UAC, DRM, and digital certificate authorities. As a result, I do my best to avoid any mention of them. I didn't mention digital certificate authorities today, but I did link to an article which mentioned them, and now the subject has overrun the comments like kudzu. I don't need to deal with this nonsense, so I'm just going to kill my <u>promised future article</u> that was related to Windows Error Reporting, (I was also planning on converting my talk on debugging application compatibility issues into future articles, but since that's also related to Windows Error Reporting, I'm going to abandon that too.)