How a glitch nearly crashed the global financial system

From Zerohedge, 10 August 2015

For all the talk of how the financial world nearly ended in the aftermath of first the Lehman bankruptcy, then the money market freeze, and culminating with the AIG bailout, and how bubble after Fed bubble has made the entire financial system as brittle as the weakest counterparty in the collateral chain of some $100 trillion in shadow liabilities, the truth is that despite all the "macroprudential" planning and preparations, all the world's credit, housing, stock, and illiquidity bubbles may be nothing when compared to the oldest "glitch" in the book: a simple cascading error which ends up taking down the entire system.

Like what happened in the great quant blow up August 2007.

For those who may not recall the specific details of how the "quant crash" nearly wiped out all algo and quant trading hedge funds and strats in a matter of hours if not minutes, leading to tens of billions in capital losses, here is a reminder, and a warning that the official goalseeked crisis narrative "after" the fact is merely there to hide the embarrassment of just how close to total collapse the global financial system is at any given moment.

The following is a true story (courtesy of b3ta) from the archives, going all the way back to 2007:

I.T. is a minefield for expensive mistakes

There's so many different ways to screw up. The best you can hope for in a support role is to be invisible. If anyone notices your support team at all, you can rest assured it's because someone has made a mistake. I've worked for three major investment banks, but at the first place I witnessed one of the most impressive mistakes I'm ever likely to see in my career. I was part of the sales and trading production support team, but thankfully it wasn't me who made this grave error of judgement...

(I'll delve into obnoxious levels of detail here to add color and context if you're interested. If not, just skip to the next chunk, you impatient git)

This bank had pioneered a process called straight-through processing (STP) which removes the normal manual processes of placement, checking, settling and clearing of trades. Trades done in the global marketplace typically have a 5-day clearing period to allow for all the paperwork and book-keeping to be done. This elaborate system allowed same-day settlement, something never previously possible. The bank had achieved this over a period of six years by developing a computer system with a degree of complexity that rivalled SkyNet. By 2006 it also probably had enough processing power to become self-aware, and the storage requirements were absolutely colossal. It consisted of hundreds of bleeding edge compute-farm blade servers, several £multi-million top-end database servers and the project had over 300 staff *just to keep it running*. To put that into perspective, the storage for this one system (one of about 500 major trading systems at the bank) represented over 80% of the total storage used within the company. The equivalent of 100 DVD's worth of raw data entered the databases each day as it handled

over a million inter-bank trades, each ranging in value from a few hundred thousand dollars to multi-billion dollar equity deals. This thing was **BIG**.

Parts of this system that no-one understood any more

You'd think such a critically important and expensive system would run on the finest, fault-tolerant hardware and software. Unfortunately, it had grown somewhat organically over the years, with bits being added here, there and everywhere. There were parts of this system that no-one understood any more, as the original, lazy developers had moved company, emigrated or *died* without documenting their work. I doubt they ever predicted the monster it would eventually become.

A colleague of mine one day decided to perform a change during the day without authorisation, which was foolish, but not uncommon. It was a trivial change to add yet more storage and he'd done it many times before so he was confident about it. The guy was only trying to be helpful to the besieged developers, who were constantly under pressure to keep the wretched thing moving as it got more bloated each day, like an electronic 'Mr Creosote'.

As my friend applied his change that morning, he triggered a bug in a notoriously crap script responsible for bringing new data disks online. The script had been coded in-house as this saved the bank about £300 per year on licensing fees for the official 'storage agents' provided by the vendor. Money that, in hindsight, would perhaps have been better spent instead of pocketed. The homebrew code took one look at the new configuration and immediately spazzed out. This monged scrap of pisspoor geek-scribble had decided the best course of action was to bring down the production end of the system and bring online the disaster recovery (DR) end, which is normal behaviour when it detects a catastrophic 'failure'. It's designed to bring up the working side of the setup as quickly as possible. Sadly, what with this system being fully-replicated at both sites (to *[cough]* ensure seamless recovery), **the exact same bug was almost instantly triggered on the DR end, so in under a minute, the hateful script had taken offline the entire system in much the same manner as chucking a spanner into a running engine might stop a car**. The databases, as always, were flushing their precious data onto many different disks as this happened, so *massive*, irreversible data corruption occurred. **That was it, the biggest computer system in the bank, maybe even the world, was down.**

And it wasn't coming back up again quickly.

(OK, detail over. Calm down)

$12 TRILLION of trades in the system

At the time this failure occurred there was more than **$12 TRILLION** of trades at various stages of the settlement process in the system. **This represented around 20% of ALL trades on the global stock market, as other banks had started to plug into this behemoth and use its capabilities themselves**. If those trades were not settled within the agreed timeframe, th**e bank would be liable for penalties on each and every one, the resulting fines would eclipse the market capital of the company, and so it would go out of business. Just like that.**

My team dropped everything it was doing and spent 4 solid, brutal hours recovering each component of the system in a desperate effort to coax the stubborn silicon back online. **After a short time, the head of the European Central Bank (ECB) was on a crisis call with our company CEO, demanding status updates as to why so many trades were failing that day**. Allegedly (as we were later told), the volume of financial goodies contained within this beast was so great that failure to clear the trades **would have had a significant negative effect on the value of the Euro currency**. This one fuckup almost started a global economic crisis on a scale similar to the recent (and ongoing) sub-prime credit crash. **With two hours to spare before the ECB would be forced to go public by adjusting the Euro exchange rate to compensate, the system was up and running, but barely.** We each manned a critical sub-component and diverted all resources into the clearing engines. The developers set the system to prioritise trades on value. Everything else on those servers was switched off to ensure every available CPU cycle and disk operation could be utilised. It saturated those machines with processing while we watched in silence, unable to influence the outcome at all.

Incredibly, the largest proportion of the high-value transactions had cleared by the close of business deadline, and disaster was averted by the most "wafer-thin" margin. Despite this, the outstanding lower-value trades still cost the bank more than $100m in fines. Amazingly, to this day only a handful of people actually understand the true source of those penalties on the end-of-year shareholder report. **Reputation is king in the world of banking and all concerned --including me-- were instructed quite explicitly to keep schtum**. Naturally, I *can't* identify the bank in question, but if you're still curious, gaz me and I'll point you in the right direction…

A pompous ceremony of blame the next day

Epilogue… The bank stumped up for proper scripts pretty quickly but the poor sap who started this ball of shit rolling was fired in a pompous ceremony of blame the next day, which was rather unfair as it was dodgy coding which had really caused the problem. The company rationale was that every blaze needs a spark to start it, and he was going to be the one they would scapegoat. That was one of the major reasons I chose to leave the company (but not before giving the global head of technology a dressing down at our Christmas party… that's another QOTW altogether). Even today my errant mate is one of the only people who properly understands most of that preposterous computer system, so he had his job back within six months -- but at a higher rate than before :-)

Conclusion: most banks are insane and they never do anything to fix problems until *after* it costs them uber-money. Did I hear you mention length? 100 million dollar bills in fines laid end-to-end is about 9,500 miles long according to Google calculator.