Malleability FAQ

Andrew Poelstra
16 February 2014

1 What's the deal with malleability?

1.1 What is transaction malleability?

Malleability is a property of a cryptosystem which describes how much an encrypted or signed message can be modified after-the-fact without invalidating the encryption or digital signature. For Bitcoin, malleability means the ability to subtly modify transactions after they have been signed.

It should be emphasized that *these modifications do not change the meaning of transactions one iota to the Bitcoin network*. Therefore, malleating a transaction cannot reroute funds or invalidate it.

Specifically, the malleability that Bitcoin transactions suffer from is caused by the fact that signed transactions include their signatures, along with a bit of metadata about the signature (its size, for example). Since Bitcoin's ECDSA signatures are unforgeable, no signed data can be changed by anyone after the fact — but since digital signatures are tied to the message they sign, it's impossible for the signature itself to be signed data!

So the parts of the transaction which have meaning to the Bitcoin network cannot be changed after signing, but the signature itself can be, in a couple of ways. This is how we know that while transactions are changed, it is impossible to change them in any serious way.

1.2 So why does this matter?

Associated to each Bitcoin transaction is the poorly-named *txid*, which identifies a transaction in the sense that no two transactions can share the same txid. txids are not stored alongside the transaction; instead they are computed by hashing up the transaction (signature and all). Therefore, if a signature is changed, even by a single bit, the txid can completely change.

This is a problem because txids are used within Bitcoin to chain transactions together. This is how Bitcoin keeps track of funds — when you spend money, you mention the transactions (by txid) in which you received that money. So because txids can change, this means that you can't spend money until your own receiving transaction has been confirmed. Then the entire network agrees on its txid and all is well.

This is not a problem for ordinary transactions, because the point of confirmations is to prevent double-spends. So if you are spending unconfirmed coins, this is already unsafe behaviour and transaction malleability does not change that fact.

It *is* a problem for some creative protocols designed to enable deposits, micropayments, certain escrow structures, etc., which depend on chaining unconfirmed transactions together and assuming nobody can break the chain. But malleability does break the chain, and these protocols have not been implemented because of this.

2 Why is this in the news?

2.1 How come all these exchanges have halted withdrawals then?

The answer to the last question suggested that this "malleability" business was an awfully esoteric problem. But there are several Bitcoin businesses who recently temporarily halted withdrawals, blaming malleability. So what gives?

Update: To the best of my knowledge, every service except Mt. Gox is back online and functioning.

With one notable exception (see the next question), the reason that Bitcoin services have shut down is that recently a Bitcoin node came online which is actually malleating transactions that come by. When this node changes the txid of a transaction, the result is two transactions (the original and the new one) which claim to send money from exactly the same receiving transactions. To the Bitcoin network, this is the definition of a double-spend.

In principle this doesn't matter — Bitcoin prevents double-spending by ensuring that only one of the two transactions will confirm, and since they're identical anyway, it doesn't matter which one. However, because of this fact, double-spends are quite rare and services accepting Bitcoin therefore become uneasy whenever they detect one. Because of this rogue node, one morning these services woke up to see tens of thousands of double-spends. As you can imagine, their systems did not like this, and they were effectively DOS (denial of service) attacked by it.

So the main reason these services shut down is to strengthen their code to handle floods of double-spends without any trouble. It is likely that they are also auditing their code to ensure that they don't depend on txids to be constant. This would be a bug, but since txids didn't change much in practice until recently, it's a safe bet that such a bug would not have been detected before now.

2.2 What about Mt. Gox?

Their troubles are entirely their own fault, and it seems like their press release was the impetus for this malleating node to start running, so everyone else's troubles are also their fault.

Their conduct and accusations were and are completely antisocial, unprofessional and simply reckless. I strongly encourage anybody storing funds with this sham of a business to get these funds out as quickly as possible.

As we will see in the next question, any money lost by Mt. Gox was lost sheerly by their own incompetence. And this is not the first time they have done so — for example, in late 2011 they sent several hundred Bitcoins to a null address[1]. Another example is that their software would try to spend immature coins (coins which were mined less than 100 blocks ago, which the network does not allow to be spent), a fact that was noticed in early September 2013 by a Bitcoin developer (*not* anyone working for Mt. Gox) when some customers were unable to withdraw their money.

[1] see transaction `aa62bdd690de061a6fbbd88420f7a7aa574ba86da4fe82edc27e2263f8743988`

2.3 Yeah but what *happened* to Mt. Gox?

Okay, one way to malleate a transaction is to change the format of its digital signature. The OpenSSL library that Bitcoin uses is not too picky about the formats that it accepts, so it is possible to tweak a transaction to have an excessively padded signature, but still have it be accepted by the network.

For a long time, at least since early 2013, Mt. Gox was exploiting this freedom of format to create transactions with excessive padding. It is not clear why they were doing this, but the nicest explanation is that this was an honest bug in their signing code. In early 2013, there was talk of cracking down on these nonstandard signatures, and it was noticed in the Bitcoin development channels that Mt. Gox was originating many of them. Of course, Mt. Gox itself claims to have been unaware of this.

As we would later learn, Mt. Gox had "no time" to notice this single fact over the course of a year, or to notice the article on the Bitcoin wiki describing this form of malleability since 2011. This is understandable, since processing Bitcoin transactions is only the entirety of their business model.

As a first step to eliminating malleability (remember, there are useful protocols which could be enabled by its elimination), in version 0.8.0 of the Bitcoin reference client poorly-padded signatures became nonstandard. This means that while transactions with invalid padding may be mined, new clients will not relay them, that is, they will play no part in actually communicating them to miners. This version was published in February 2013.

With this change, Mt. Gox's nonstandard transactions suddenly took a long time to get through the network, and confirmations were very slow. At this point, Mt. Gox really had no excuse not to realize their mistake, since it was causing very user-visible problems and the explanation was being mentioned on IRC at least once every single day.

A few enterprising users realized that malleability was their saviour: by connecting directly to Mt. Gox, they could find their withdrawals, fix the signature themselves, and then relay it. By doing this, these users were able to process their withdrawals quickly despite Gox's incompetence. A bit later, they noticed something else: when they changed the transaction, they'd change the txid, and this meant that Mt. Gox's website would not notice the transaction going through.

To be clear: Mt. Gox was sending money on the Bitcoin network and was *not keeping track of this fact*. This is money-losing mistake number one on their part.

Next, because Mt. Gox would not notice the withdrawals, the customer could request that the withdrawal be reprocessed, and Gox would automatically do so. Now, in Bitcoin there is a notion of "resending" money. That is, if a transaction is not confirmed, you can create a new transaction with the same inputs (but with perhaps a higher fee, or a properly formed signature, say). This is a double-spend, so you can be assured that at most one of the transactions will go through. In this way you can resend money without fear of somehow spending twice as much as you meant to.

If you don't respend money in this way, **you will get robbed**. This should be common sense to anybody doing payment processing, and cannot be emphasized enough. An analogy to traditional banking would be if Gox was reissuing cheques without cancelling the old ones.

Rather than resending money, Mt. Gox chose to send *different* money. That is, their response

to losing track of their money was to automatically spend even more, to anyone who requested, and still not track these spends. This is money-losing mistake number two.

On February 10, 2014, Mt. Gox finally noticed this problem. (Perhaps an angel appeared to Mark Karpeles/MagicalTux in a dream, or more likely they listened to one of their customers, who explained this problem to them. Again, Mt. Gox was unable to notice this on their own). Rather than asking the Bitcoin developers for advice, or fixing their own software, or asking any informed person to proofread, they published without warning a press release which claimed there was a "flaw in the Bitcoin protocol" called "transaction malleability" and that the Bitcoin developers were at fault for their losing money. Furthermore, they halted withdrawals and demanded without pay a "fix" from the Bitcoin developers before they would re-open.

Now, ignoring for a second the lies and extortion here, this request makes no sense at all. Firstly, Mt. Gox's code has nothing to do with code written by the Bitcoin developers. Secondly, the two ways that Gox lost money (not tracking transactions and "resending" by sending even more money) have nothing to do with transaction malleability.

The next day, the malleating node went online, no doubt spurred by this press release, and caused the other exchanges to be DOSed.

2.4 What about the Silk Road 2?

SR2's statement appears to be a very clumsy and transparent lie. They provided no details and still managed to contradict reality (claiming there is a "bitcoin exploit" that allows theft of funds), so that's the best response I can give.

3 So what is being done about this?

3.1 Is there something wrong with the Bitcoin client?

There are two problems with the reference Bitcoin client which have come to light amidst this malleability story, and neither of them allow funds to be lost or stolen.

The first is that the Bitcoin client does not handle double-spends with any grace, and this includes "double-spends" which are actually malleated transactions. This has been a known problem, but not a high-priority one, since it's impossible to create double-spends with the Bitcoin client anyway. The consequence of this is that the client's transaction list will show both sides of a double-spend, even though only one can be confirmed. The other will remain in the client indefinitely, showing zero confirmations and taking up space. This can also cause the available balance display to be incorrect. To be clear, this is purely a display bug; there is no loss (or even potential loss) of funds.

The second problem is that the Bitcoin client is willing to spend its own change without confirmations, since change is money sent to the client from itself. (The risk with unconfirmed transactions is that the other party will behave in an untrustworthy way. For change, both parties are the client itself, and it trusts itself.) However, if the txid of the change transaction changes, any new transactions which spend this change will become invalid. Again, the result is a never-confirming

transaction taking up space, though this one is more problematic because it also prevents you from actually spending the money, as long as it thinks the invalid transaction already has.

The solution to both problems is to (a) detect and handle malleated or double-spent transactions properly, and (b) give the user the ability to remove unconfirmed transactions from the wallet.

The current flurry of activity on the Bitcoin development channel and on Github is entirely related to these fixes.

3.2 What about eliminating malleability?

As was mentioned, the problem of malleability has been known since at least 2011. Since it has prevented the development of new protocols, why hasn't it been fixed?

Well, one reason is that it's a fairly low-priority change and would require changes to the consensus code, which is a huge hairy deal to do while preventing divisive "forks" where the network is unable to agree on which rules are in play for validating transactions.

There is a deeper reason that this hasn't been fixed though, which is a bit technical. In order to enable protocols which chain unconfirmed transactions together, we need to eliminate all possible means of malleating transactions, not just the ones that we're aware of today. This means that we need to make sure not only that signatures are encoded consistently, but that the signatures themselves cannot be changed.

Bitcoin signatures consist of a pair of numbers (the exact algorithm is ECDSA, which you can look up on Wikipedia) which are required to satisfy an algebraic identity which involves the signing key of the signer as well as a hash of the message to be signed. It is an open problem whether or not there are any algebraic manipulations which can change (r,s) but still pass this validation equation. Such a manipulation would be a very clever form of malleability, but a form of malleability nonetheless.

Proving mathematically that this can't be done has shown to be a very difficult problem, and that is what is necessary to fully eliminate malleability. Until this is done, we cannot grant ourselves the freedom of nonmalleability, and so progress has been low-priority and slow.