Financial fraud happens when someone deprives you of your money or otherwise harms your **financial** health through misleading, deceptive, or other illegal practices. This can be done through a variety of methods such as identity theft or investment **fraud**.

It is important to report the crimes to the appropriate agencies and law enforcement as soon as possible irrespective of the type of financial fraud. Fraudulent charges should also be disputed or cancelled as soon as they are discovered as well. Furthermore, victims should collect documentation related to the crime, such as bank statements, credit reports, tax form from current and previous years, and continue to file important information throughout the reporting process.

Types of Financial Crimes:

1. **Identity Theft:* *Someone steals your personal financial information, such as credit card number or bank account number, to make fraudulent withdrawals from your account. Sometimes people will use the information to open credit or bank accounts and leave the victim liable for all the charges. Identity theft may lead to damaged credit rating, bounced checks/denied payments, and being pursued by collection agencies.
2. **Investment Fraud:** This type includes selling investments or securities with false, misleading information. It could be false promises, hiding facts, and insider trading tips.
3. **Mortgage and Lending Fraud:** ****A third-party may open a mortgage or loan using your information or using false information. In another case, lenders may sell mortgage or loans with inaccurate information, deceptive practices, and other high pressure sales tactics. 4.*** *Mass Marketing Fraud:** The fraud is committed through mass mailings, telephone calls, or spam emails. It also includes fake checks, charities, lotteries, honor society invitations, and more. These modes are used to steal personal financial information or to raise contributions to fraudulent organisations.

Common Web Security Mistake #1: Injection flaws

Injection flaws result from a classic failure to filter untrusted input. It can happen when you pass unfiltered data to the SQL server (SQL injection), to the browser (XSS – we'll talk about this later), to the LDAP server (LDAP injection), or anywhere else. The problem here is that the attacker can inject commands to these entities, resulting in loss of data and hijacking clients' browsers.

Anything that your application receives from untrusted sources must be filtered, preferably according to a whitelist. You should almost never use a blacklist, as getting that right is very hard and usually easy to bypass. Antivirus software products typically provide stellar examples of failing blacklists. Pattern matching does not work.

Prevention: The good news is that protecting against injection is "simply" a matter of filtering your input properly and thinking about whether an input can be trusted. But the bad news is that *all* input needs to be properly filtered, unless it can unquestionably be trusted (but the saying "never say never" does come to mind here).

In a system with 1,000 inputs, for example, successfully filtering 999 of them is not sufficient, as this still leaves one field that can serve as the Achilles heal to bring down your system. And you might think that putting an SQL query result into another query is a good idea, as the database is trusted, but if the perimeter is not, the input comes indirectly from guys with malintent. This is called Second Order SQL Injection in case you're interested.

Since filtering is pretty hard to do right (like crypto), what I usually advise is to rely on your framework's filtering functions: they are proven to work and are thoroughly scrutinized. If you do not use frameworks, you really need to think hard about whether *not* using them really makes sense in your server security context. 99% of the time it does not.

Common Web Security Mistake #2: Broken Authentication

This is a collection of multiple problems that might occur during broken authentication, but they don't all stem from the same root cause.

Assuming that anyone still wants to roll their own authentication code in 2014 (what are you thinking??), I advise against it. It is extremely hard to get right, and there are a myriad of possible pitfalls, just to mention a few:

1. he URL might contain the session id and leak it in the referer header to someone else.
2. The passwords might not be encrypted either in storage or transit.
3. The session ids might be predictable, thus gaining access is trivial.
4. Session fixation might be possible.
5. Session hijacking might be possible, timeouts not implemented right or using HTTP (no SSL security), etc…

Prevention: The most straightforward way to avoid this web security vulnerability is to use a framework. You might be able to implement this correctly, but the former is much easier. In case you do want to roll your own

code, be extremely paranoid and educate yourself on what the pitfalls are. There are quite a few.

Common Web Security Mistake #3: Cross Site Scripting (XSS)

This is a fairly widespread input sanitization failure (essentially a special case of common mistake #1). An attacker gives your web application JavaScript tags on input. When this input is returned to the user unsanitized, the user's browser will execute it. It can be as simple as crafting a link and persuading a user to click it, or it can be something much more sinister. On page load the script runs and, for example, can be used to post your cookies to the attacker.

Prevention: There's a simple web security solution: don't return HTML tags to the client. This has the added benefit of defending against HTML injection, a similar attack whereby the attacker injects plain HTML content (such as images or loud invisible flash players) – not high-impact but surely annoying ("please make it stop!"). Usually, the workaround is simply converting all HTML entities.

A **computer virus** is a malicious software program loaded onto a user's **computer** without the user's knowledge and performs malicious actions. ... It can self-replicate, inserting itself onto other programs or files, infecting them in the process. Not all **computer viruses** are destructive though.

"Malware" encompasses **computer viruses** along with many other forms of malicious software, such as **computer** "worms", ransomware, spyware, adware, trojan horses, keyloggers, rootkits, bootkits, malicious Browser Helper Object (BHOs), and other malicious software.

In general usage, the term "**computer virus**" includes all forms of "malware," or malicious software. Instead of sniffles and a fever, some common symptoms of a **computer** viral infection are slow performance, data loss and system crashes, all of which can make people using the machine feel ill as well.