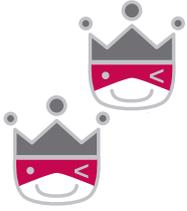


2

Spoofing

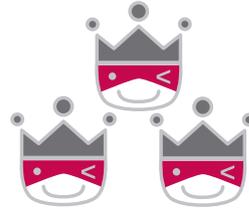
An attacker could squat on the random port or socket that the server normally uses



3

Spoofing

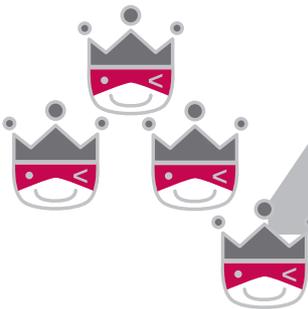
An attacker could try one credential after another and there's nothing to slow them down (online or offline)



4

Spoofing

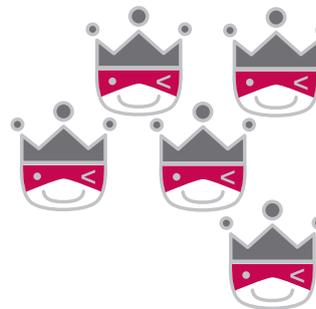
An attacker can anonymously connect, because we expect authentication to be done at a higher level



5

Spoofing

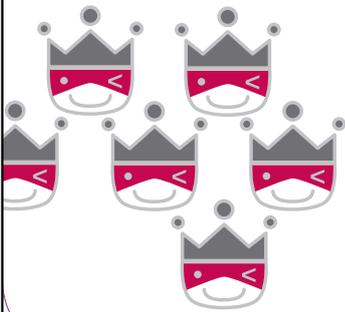
An attacker can confuse a client because there are too many ways to identify a server



6

Spoofing

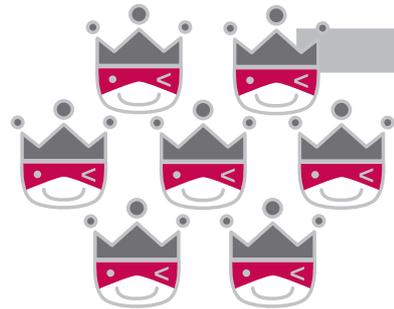
An attacker can spoof a server because identifiers aren't stored on the client and checked for consistency on re-connection (that is, there's no key persistence)



7

Spoofing

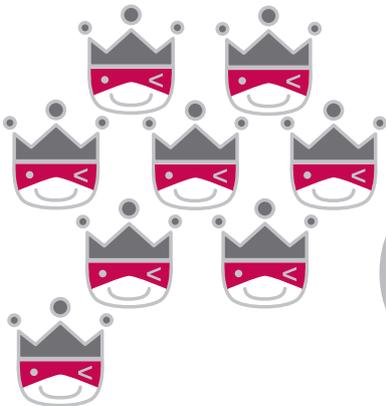
An attacker can connect to a server or peer over a link that isn't authenticated (and encrypted)



8

Spoofing

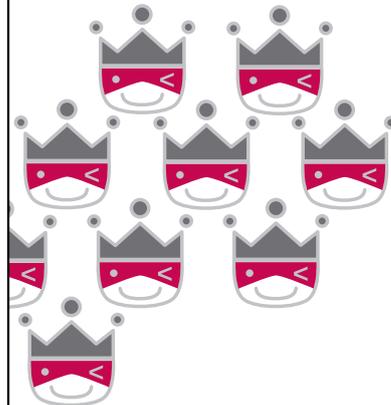
An attacker could steal credentials stored on the server and reuse them (for example, a key is stored in a world readable file)



9

Spoofing

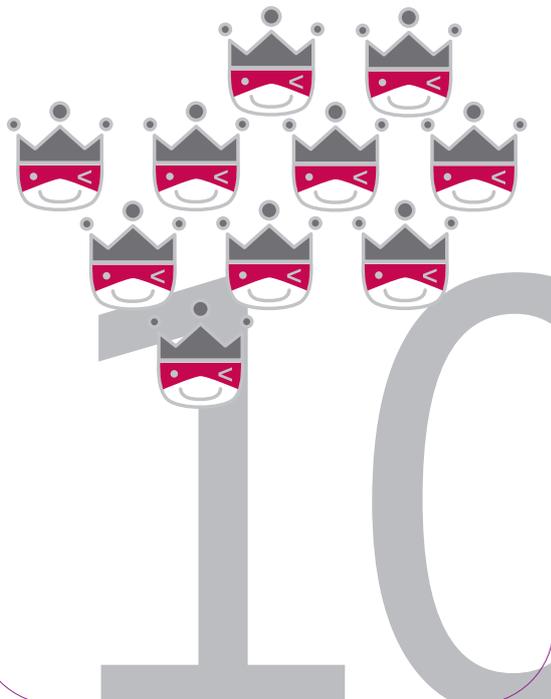
An attacker who gets a password can reuse it (Use stronger authenticators)



10

Spoofing

An attacker can choose to use weaker or no authentication



J

Spoofing

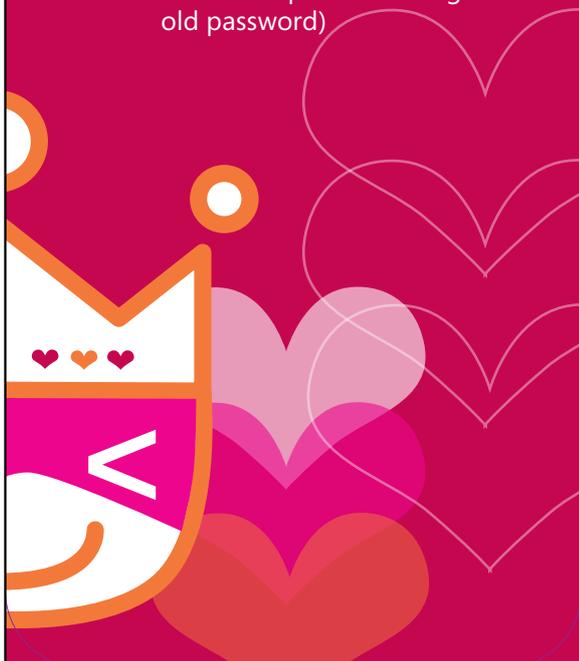
An attacker could steal credentials stored on the client and reuse them



Q

Spoofing

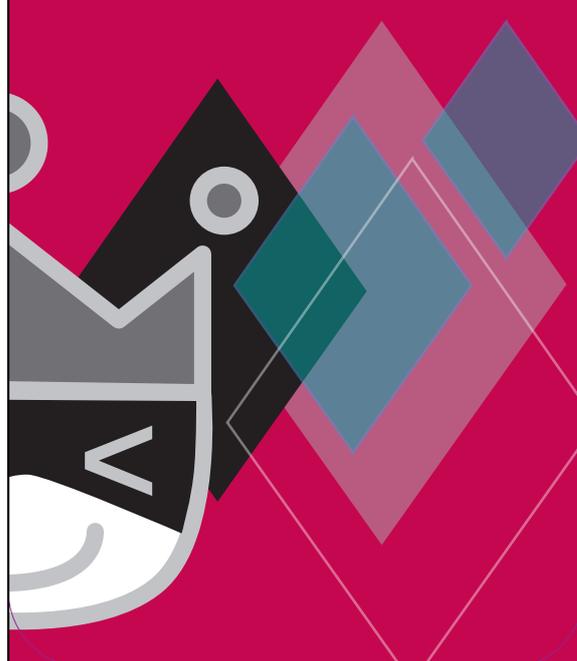
An attacker could go after the way credentials are updated or recovered (account recovery doesn't require disclosing the old password)



K

Spoofing

Your system ships with a default admin password, and doesn't force a change



A

Spoofting

You've invented a new Spoofting attack



3

Tampering

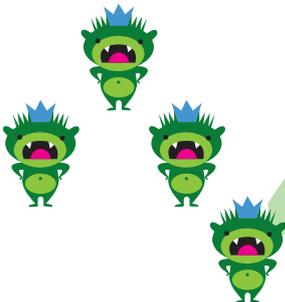
An attacker can take advantage of your custom key exchange or integrity control which you built instead of using standard crypto



4

Tampering

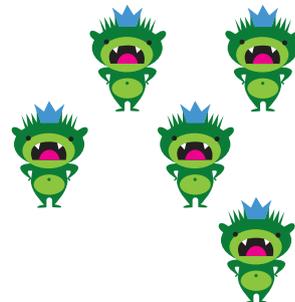
Your code makes access control decisions all over the place, rather than with a security kernel



5

Tampering

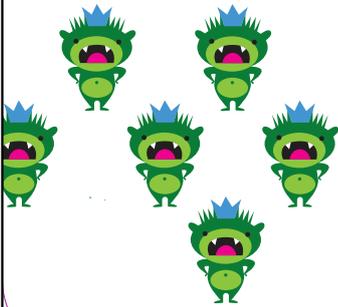
An attacker can replay data without detection because your code doesn't provide timestamps or sequence numbers



6

Tampering

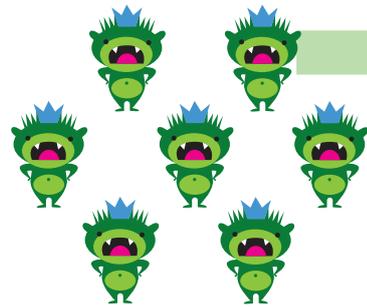
An attacker can write to a data store your code relies on



7

Tampering

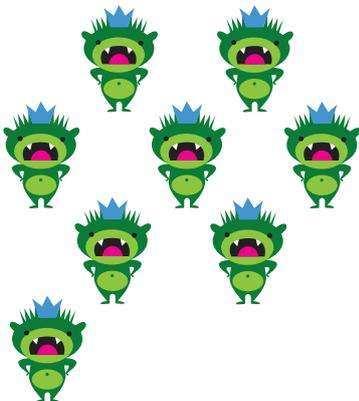
An attacker can bypass permissions because you don't make names canonical before checking access permissions



8

Tampering

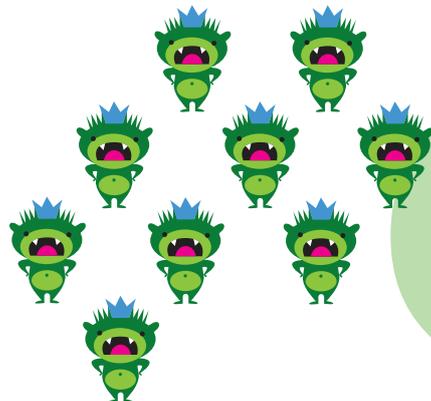
An attacker can manipulate data because there's no integrity protection for data on the network



9

Tampering

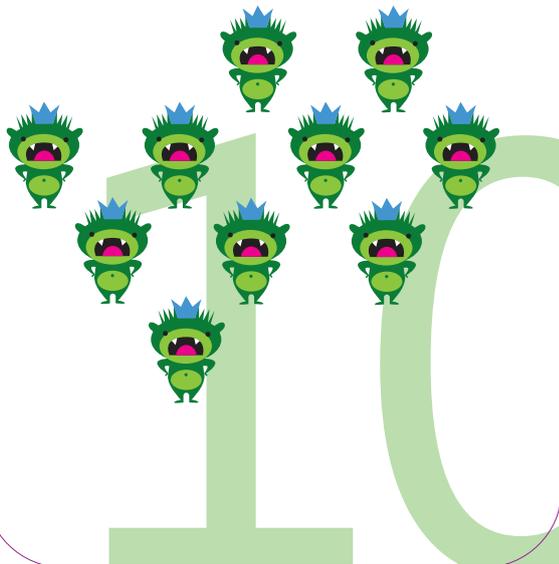
An attacker can provide or control state information



10

Tampering

An attacker can alter information in a data store because it has weak ACLs or includes a group which is equivalent to everyone ("all Live ID holders")



J

Tampering

An attacker can write to some resource because permissions are granted to the world or there are no ACLs



Q

Tampering

An attacker can change parameters over a trust boundary and after validation (for example, important parameters in a hidden field in HTML, or passing a pointer to critical memory)



K

Tampering

An attacker can load code inside your process via an extension point



A

Tampering

You've invented a new Tampering attack



2

Repudiation

An attacker can pass data through the log to attack a log reader, and there's no documentation of what sorts of validation are done



3

Repudiation

A low privilege attacker can read interesting security information in the logs



4

Repudiation

An attacker can alter digital signatures because the digital signature system you're implementing is weak, or uses MACs where it should use a signature



5

Repudiation

An attacker can alter log messages on a network because they lack strong integrity controls



6

Repudiation

An attacker can create a log entry without a timestamp (or no log entry is timestamped)



7

Repudiation

An attacker can make the logs wrap around and lose data



8

Repudiation

An attacker can make a log lose or confuse security information



9

Repudiation

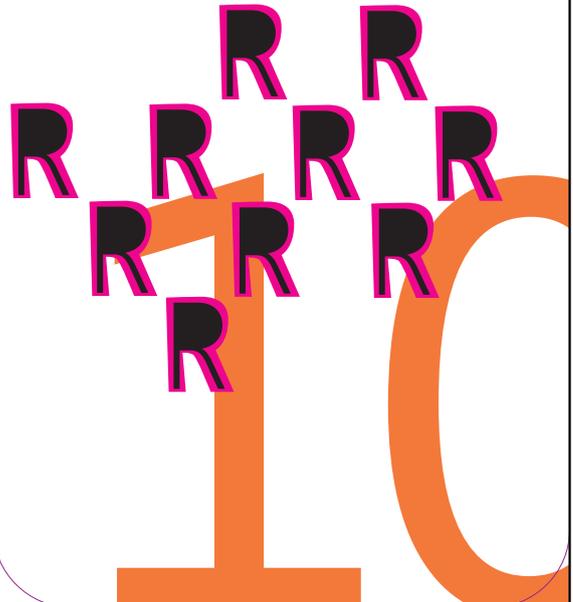
An attacker can use a shared key to authenticate as different principals, confusing the information in the logs



10

Repudiation

An attacker can get arbitrary data into logs from unauthenticated (or weakly authenticated) outsiders without validation



J

Repudiation

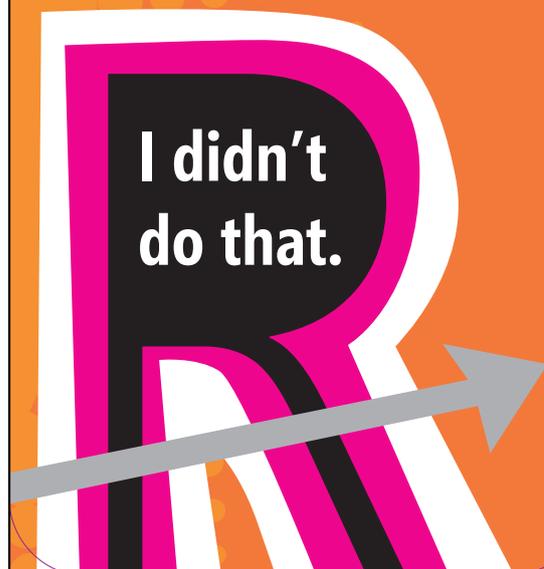
An attacker can edit logs and there's no way to tell (perhaps because there's no heartbeat option for the logging system)



Q

Repudiation

An attacker can say "I didn't do that," and you'd have no way to prove them wrong



K

Repudiation

The system has no logs

logs = 0

A

Repudiation

You've invented a new
Repudiation attack

R

A

2

Information Disclosure

An attacker can brute-force file encryption because there's no defense in place (example defense: password stretching)



3

Information Disclosure

An attacker can see error messages with security sensitive content



4

Information Disclosure

An attacker can read content because messages (say, an email or HTTP cookie) aren't encrypted even if the channel is encrypted



5

Information Disclosure

An attacker may be able to read a document or data because it's encrypted with a non-standard algorithm



6

Information Disclosure

An attacker can read data because it's hidden or occluded (for undo or change tracking) and the user might forget that it's there



7

Information Disclosure

An attacker can act as a 'man in the middle' because you don't authenticate endpoints of a network connection



8

Information Disclosure

An attacker can access information through a search indexer, logger, or other such mechanism



9

Information Disclosure

An attacker can read sensitive information in a file with bad ACLs



10

Information Disclosure

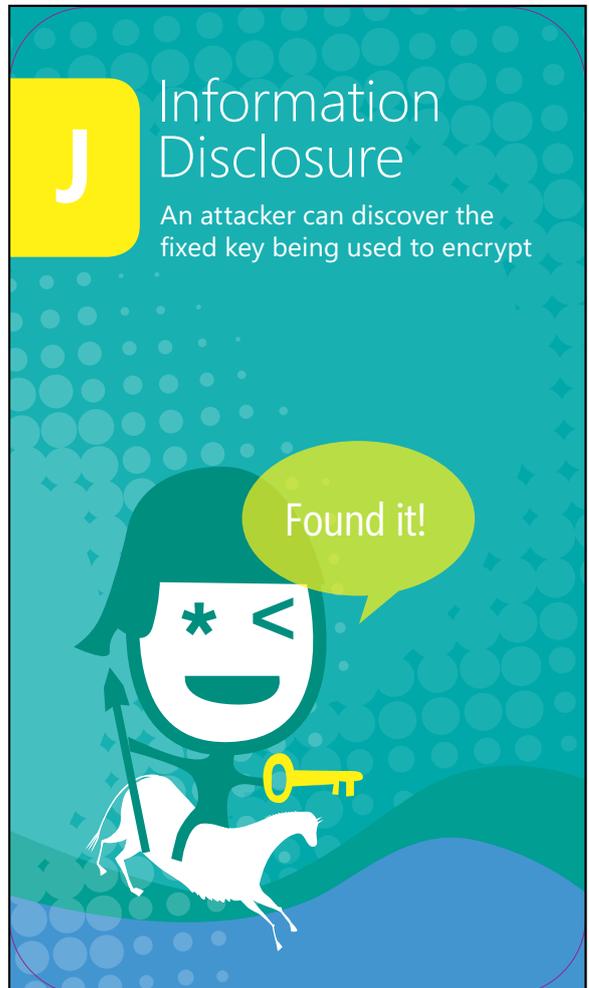
An attacker can read information in files with no ACLs



J

Information Disclosure

An attacker can discover the fixed key being used to encrypt



Q

Information Disclosure

An attacker can read the entire channel because the channel (say, HTTP or SMTP) isn't encrypted

Don't tell anyone, but...

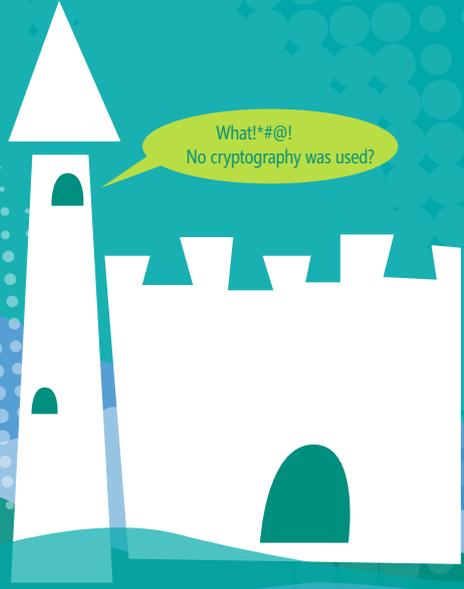


K

Information Disclosure

An attacker can read network information because there's no cryptography used

What! *#@!
No cryptography was used?



A

Information Disclosure

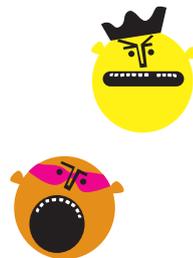
You've invented a new Information Disclosure attack



2

Denial of Service

An attacker can make your authentication system unusable or unavailable



3

Denial of Service

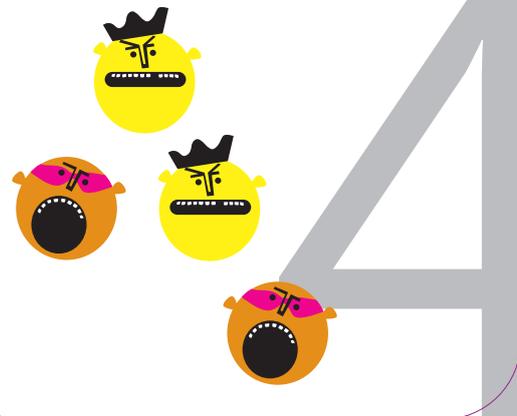
An attacker can make a client unavailable or unusable but the problem goes away when the attacker stops



4

Denial of Service

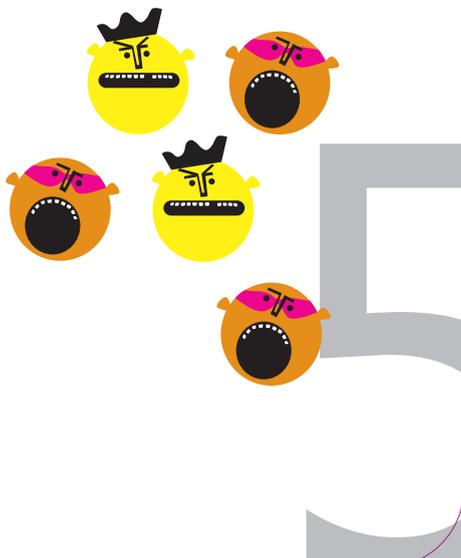
An attacker can make a server unavailable or unusable but the problem goes away when the attacker stops



5

Denial of Service

An attacker can make a client unavailable or unusable without ever authenticating but the problem goes away when the attacker stops



6

Denial of Service

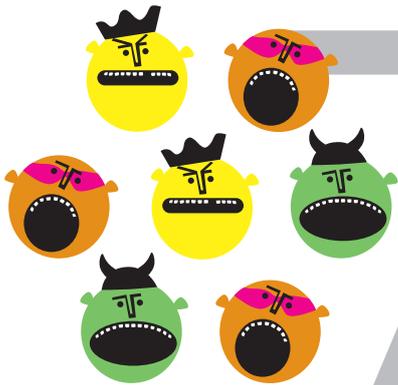
An attacker can make a server unavailable or unusable without ever authenticating but the problem goes away when the attacker stops



7

Denial of Service

An attacker can make a client unavailable or unusable and the problem persists after the attacker goes away



8

Denial of Service

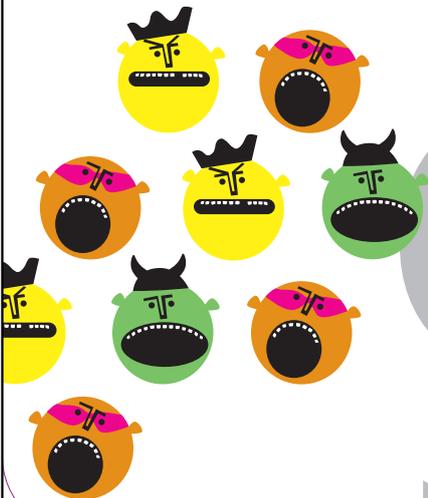
An attacker can make a server unavailable or unusable and the problem persists after the attacker goes away



9

Denial of Service

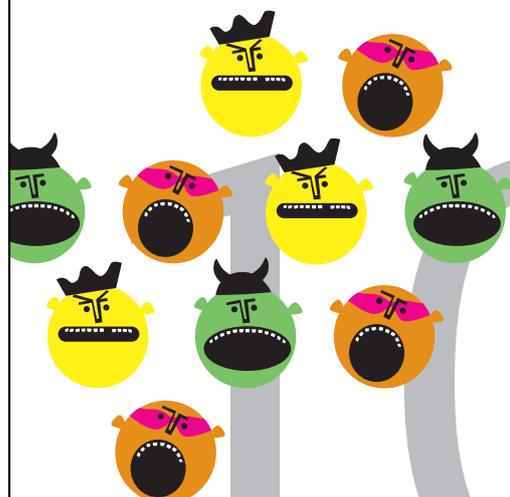
An attacker can make a client unavailable or unusable without ever authenticating and the problem persists after the attacker goes away



10

Denial of Service

An attacker can make a server unavailable or unusable without ever authenticating and the problem persists after the attacker goes away



J

Denial of Service

An attacker can cause the logging subsystem to stop working



Q

Denial of Service

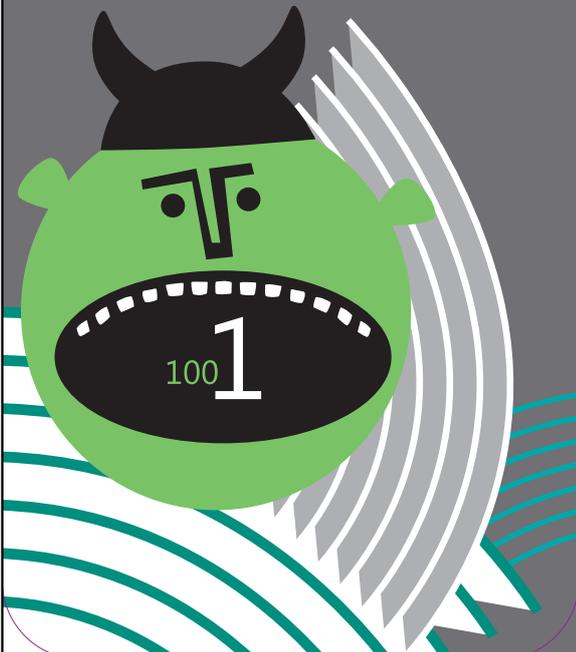
An attacker can amplify a Denial of Service attack through this component with amplification on the order of 10:1



K

Denial of Service

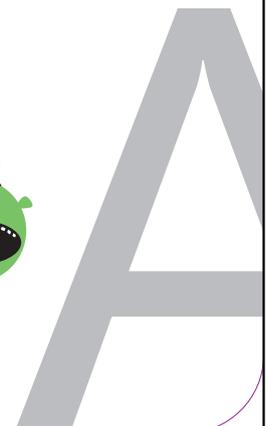
An attacker can amplify a Denial of Service attack through this component with amplification on the order of 100:1



A

Denial of Service

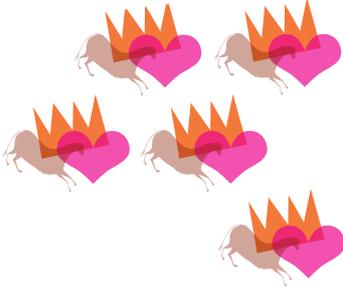
You've invented a new Denial of Service attack



5

Elevation of Privilege

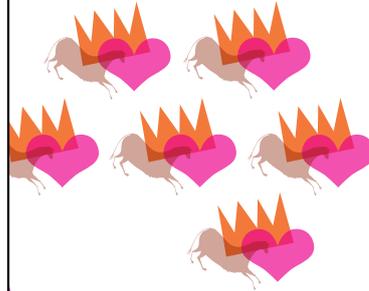
An attacker can force data through different validation paths which give different results



6

Elevation of Privilege

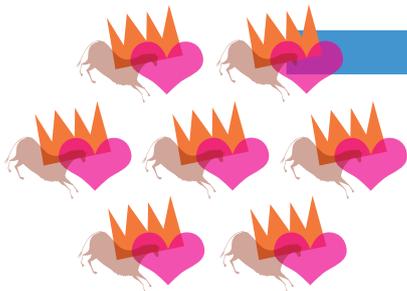
An attacker could take advantage of .NET permissions you ask for, but don't use



7

Elevation of Privilege

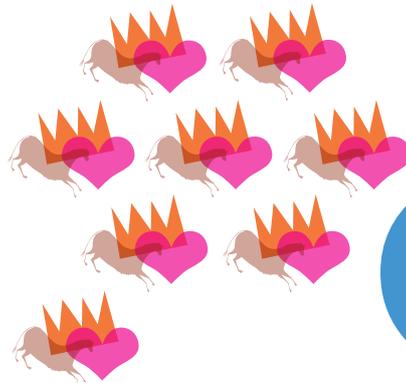
An attacker can provide a pointer across a trust boundary, rather than data which can be validated



8

Elevation of Privilege

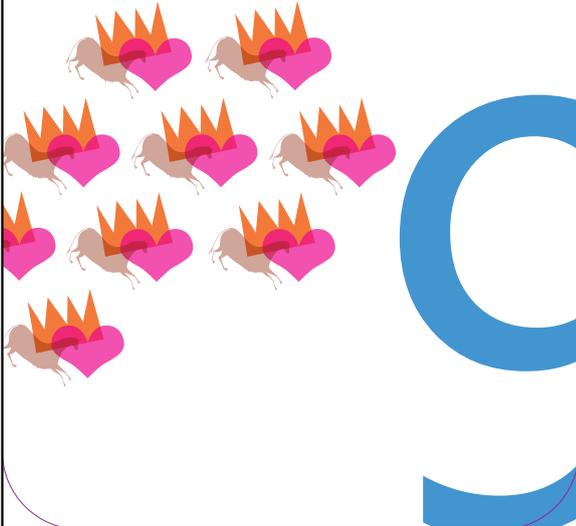
An attacker can enter data that is checked while still under their control and used later on the other side of a trust boundary



9

Elevation of Privilege

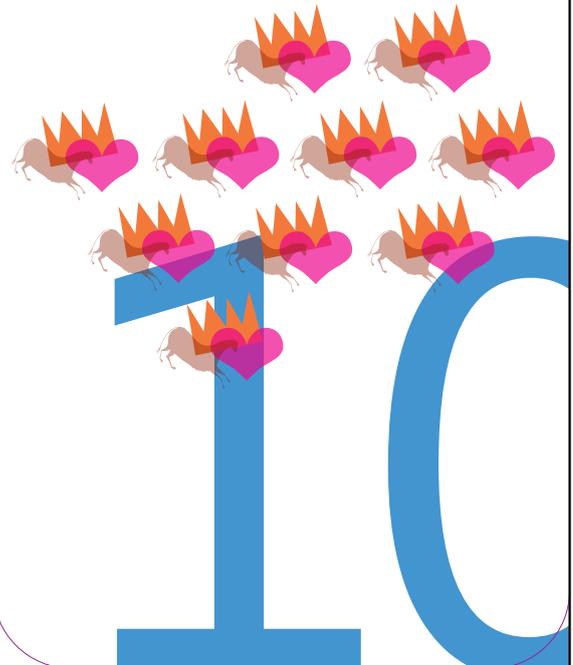
There's no reasonable way for a caller to figure out what validation of tainted data you perform before passing it to them



10

Elevation of Privilege

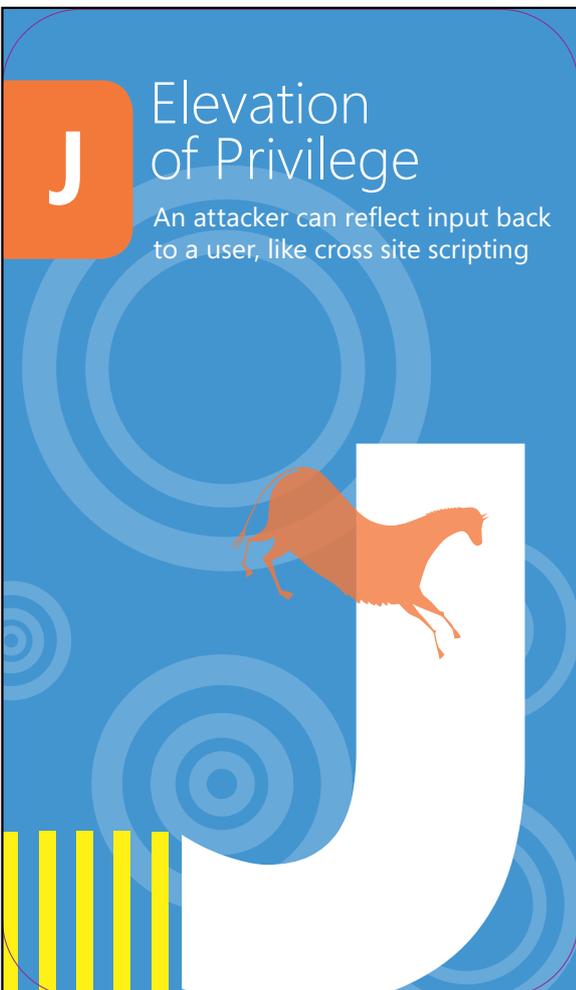
There's no reasonable way for a caller to figure out what security assumptions you make



J

Elevation of Privilege

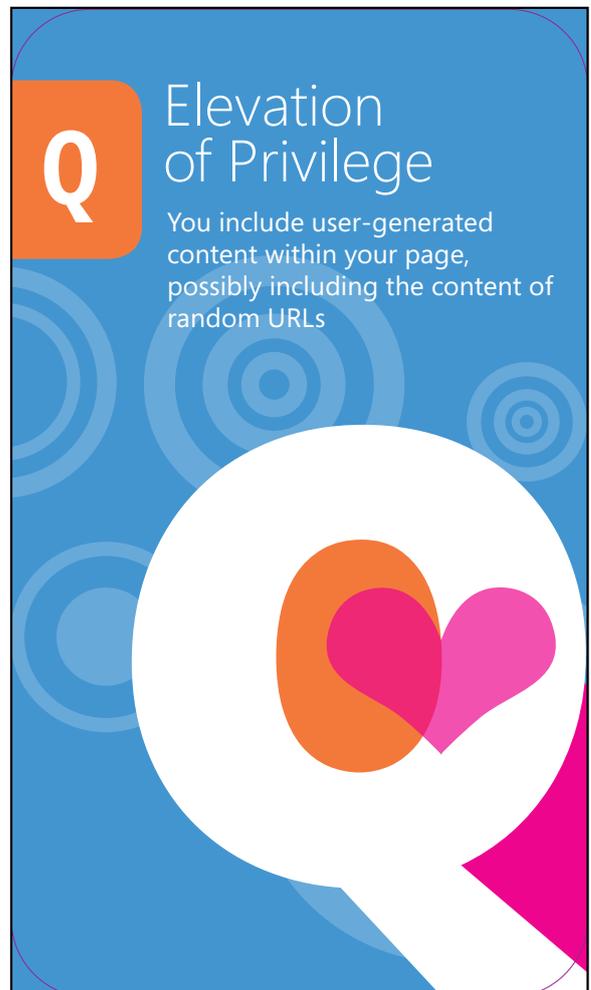
An attacker can reflect input back to a user, like cross site scripting



Q

Elevation of Privilege

You include user-generated content within your page, possibly including the content of random URLs



K

Elevation of Privilege

An attacker can inject a command that the system will run at a higher privilege level



A

Elevation of Privilege

You've invented a new Elevation of Privilege attack

