How Intigriti keeps your data safe with application-level encryption

BY INTIGRITI · MARCH 23, 2025 · LAST UPDATED ON JANUARY 2, 2026

What you will learn

- **Why application level encryption matters beyond HTTPS and basic transport level protections,** including the specific threats it mitigates (like storage attacks and root certificate interception) for sensitive platform data.
- **How Intigriti implements multi layered encryption to protect customer and researcher data from the moment it enters the system**, using rotating master keys, subkeys, and hardware backed security modules.
- **How responsible data handling and secure deletion practices support compliance and safeguard privacy,** making sure that encrypted data becomes irrecoverable once keys are destroyed.

Our application-level encryption process is unmatched by any other bug bounty platform.

At Intigriti, we know how important it is for our customers to keep their data safe. After all, bug bounty and crowdsourced security platforms handle a wealth of sensitive information, including vulnerability submissions, researcher communications, and financial data.

However, it's not only our customers' data that we need to keep safe but also that of our global community of ethical hackers. It's imperative that we don't disclose any personally identifiable information that might tie their username to their real identity.

Most bug bounty platforms are doing a satisfactory job when it comes to data protection, but Intigriti takes things a step further and uses **application-level encryption** to ensure maximum protection for our customers and our researchers.

Covering the basics

As cybersecurity service providers, all bug bounty platforms are aware of their own responsibilities to keep their client's data safe. And for the most part, they are doing a good job.

All the major crowdsourced security platforms make use of the HTTPS protocol for communications and data transfer over the web.

YOU MIGHT ALSO LIKE Intigriti obtains SOC 2 certification

HTTPS (Hypertext Transfer Protocol Secure) is an extension of HTTP, where the communication protocol is encrypted using Transport Layer Security (TLS). This significantly improved the security of the web communications and is now used more often by users than the original, non-secure HTTP.

Despite its advantages, however, HTTPS is often mistakenly referred to as ‘end-to-end encryption’. Although Intigriti also makes use of this protocol, we do not solely rely on it for our security.

Where HTTPS falls short

HTTPS is a great way of authenticating web communication and protecting the privacy and integrity of exchanged data. However, the protocol cannot be classed as ‘end-to-end encryption’ because it falls short in several areas, including:

Root certificate attacks

HTTPS does not protect your data in the event of a root certificate being installed on your device. These certificates could allow governments or malicious actors to intercept, decrypt, and re-encrypt any traffic passing through systems under their control.

Data theft

HTTPS encrypts data in transit but does not prevent malicious third parties from obtaining data at rest if they have the necessary credentials or knowhow.

Storage attacks

In case the physical storage is targeted by attackers (being disks stolen in the data center or the database exploited), attackers would be able to steal data.

To protect against these attacks, data must be encrypted at the application level – something Intigriti does as standard.

Application-level encryption by default

In addition to HTTPS encryption for web communications, Intigriti encrypts all customer and researcher data at the application level.

Intigriti’s application-level encryption ensures all submission data is encrypted from the moment it enters our servers.

A 512-bit master key (rotated every 30 days) is used to generate subkeys for every specific use. We also use NIST’s Recommendation for Key Derivation Using Pseudorandom Functions (PDF).

Importantly, this security mechanism doesn’t just apply to our bug bounty customers, but all our clients, whether this is bug bounty, managed VDP, or PTaaS

Multi-layered protection

Every security vulnerability submitted to the Intigriti platform is encrypted on multiple levels:

- Submission
- Company

- User

We have encrypted variants of the submission key for every researcher. In practise, when it comes to decrypting and viewing the submission, the following process takes place:

```
Encrypted SubmissionKey > Retrieve UserKey > Decrypt SubmissionKey > Decrypt content
```

For our customers, we have another variant of the submission key, which is encrypted with the company key. Every company member has a copy of that company key that needs to be decrypted with their user ID:

```
Encrypted SubmissionKey > Retrieve encrypted CompanyKey > Retrieve UserKey > Decrypt CompanyKey > Decrypt SubmissionKey > Decrypt submission content
```

For our database, we use the FIPS 140-2-certified Google Cloud Hardware Security Module (HSM) for encryption. To decrypt our 512-bit root key, an assailant would need to be able to steal the secrets embedded within this module.

To do this, they would need to have cloud environment access to the HSM or take over the entire Google HSM service. Even then, an attacker would still need to obtain the submission keys that are stored in our database to decrypt the content.

Responsible data handling

At Intigriti, we take our responsibility in keeping customer data secure seriously, and our application-level encryption sets us apart from the competition.

And when it comes to the deletion of your data for compliance purposes, our data destruction process is quick and effective, because once the encryption keys are deleted, the customer data is rendered useless.

Niels Hofmans, Intigriti's head of security, explains: "By encrypting customer data on top of the encryption-at-rest done by the storage provider and the encryption-in-transit by HTTPS, we ensure data compartmentalization on a multi-tenant system is done in a secure and efficient way to ensure our customers' most confidential information is appropriately handled."

READ MORE Empowering hackers through bug bounty and crowdsourced security