Bits on Bitcoin

a twist on verifying software releases

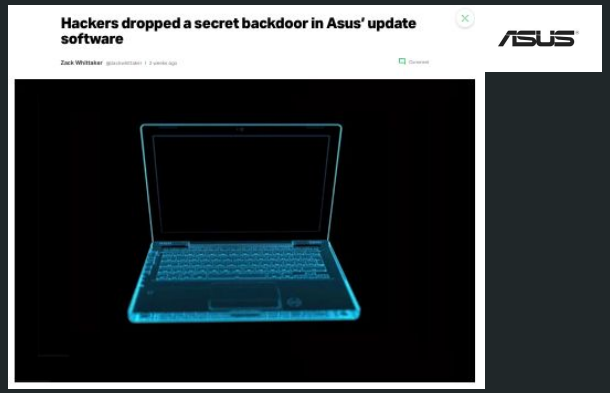
Justin Moore
Demo Day - April 9, 2019
Insight Decentralized Consensus ‘19

The cybersecurity industry is booming!

$3.5 billion industry in 2004 to
$120 billion industry in 2017

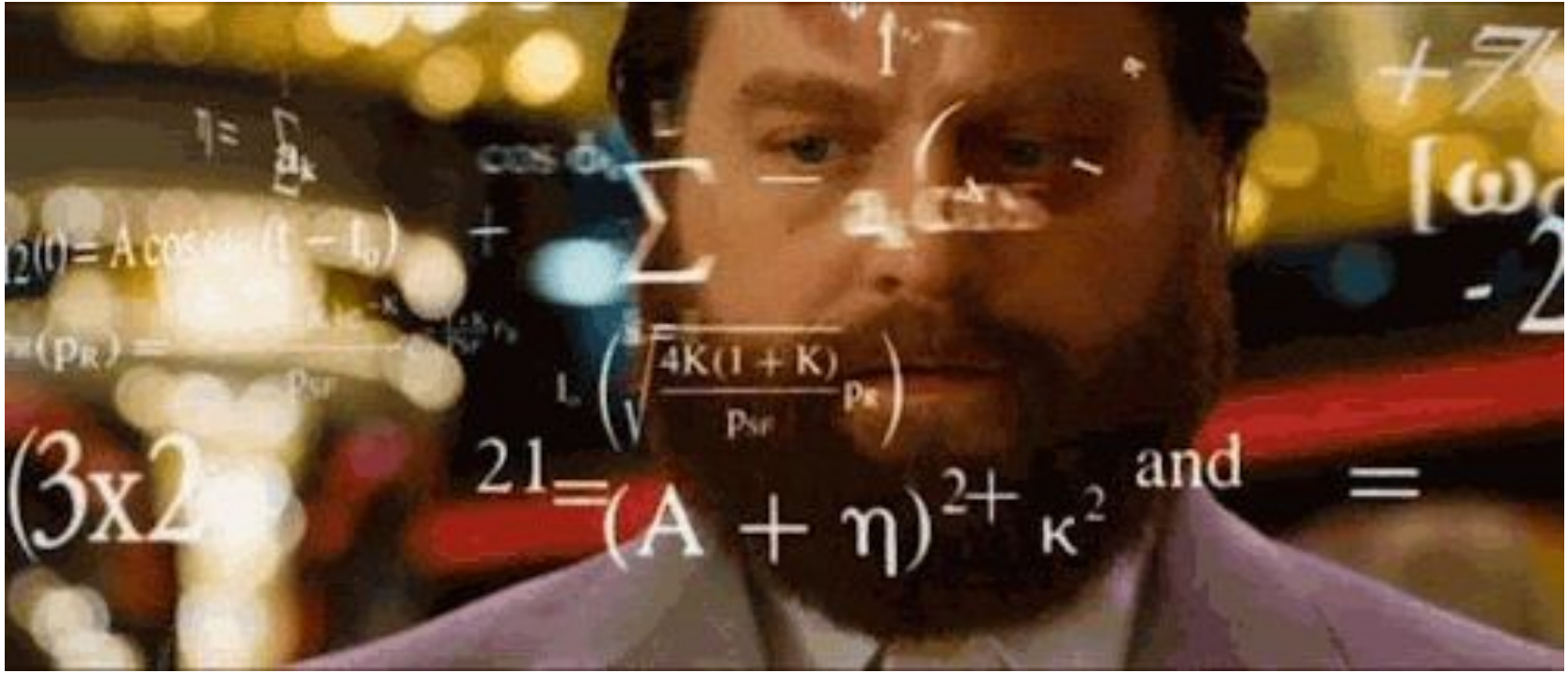
35X in 13 years.

https://cybersecurityventures.com/cybersecurity-market-report/



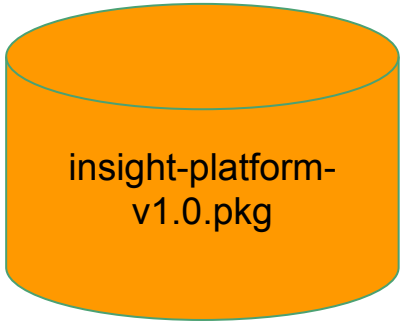
But breaches, leaks, and fraud are booming too…

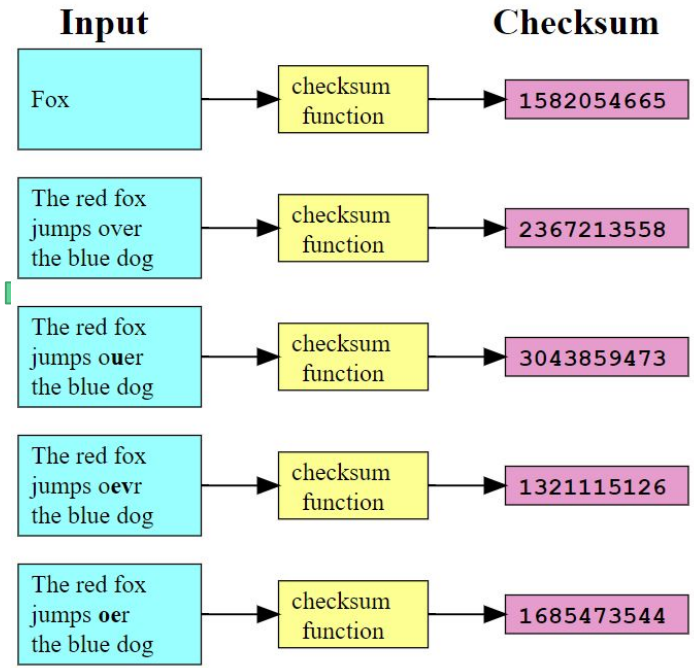
Wait, what? Something doesn’t add up.



Maybe we need to build on a new foundation.

Generate a checksum of your project

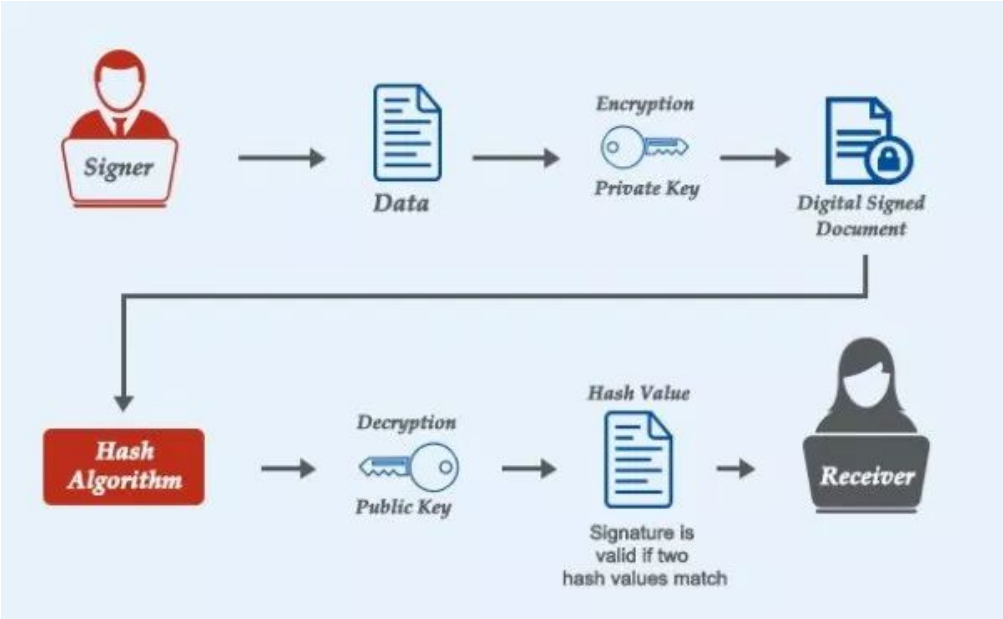




f178ed7f1715
4b292f7bc06
aba2e35dcfb
467d39

So this provides data integrity,
but what does it say about data security?

Asymmetric Cryptography and Digital Signatures



There’s nothing wrong with this approach.

...or is there?

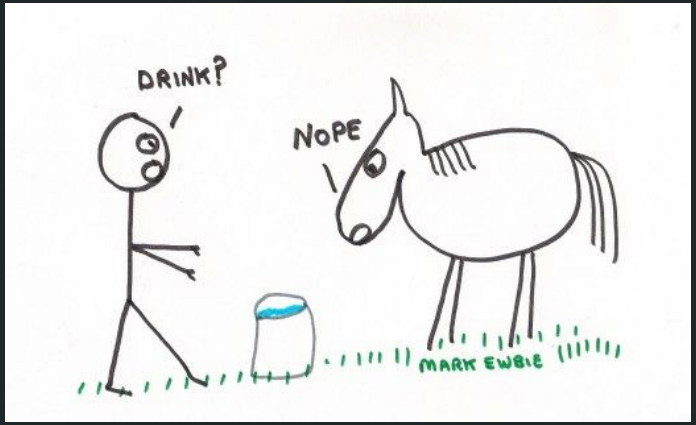
How many of you have verified <u>both</u>

- ☑ a check/shasum
- ☑ a gpg signature

<u>before</u> installing an application?

Right.

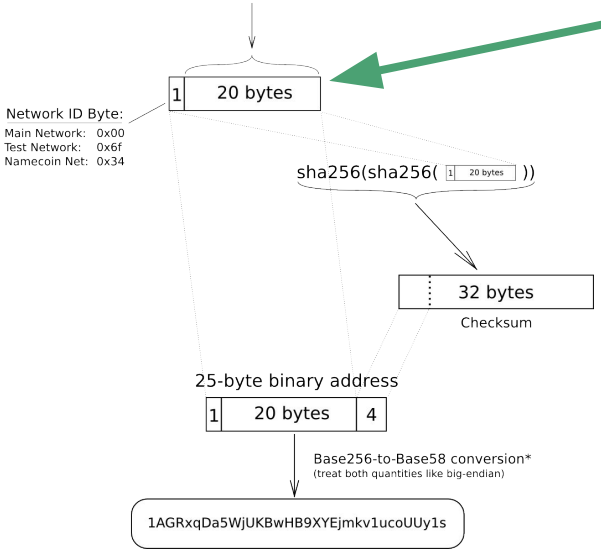
So we have a security mechanism in place, but hardly anyone actually performs the operations.



We mostly think of Bitcoin addresses as encoded outputs from public keys, but they can also be seen as information vehicles.

ECC PubKey to BTC 1addr

We could write whatever we want in these 20 bytes and the process of encoding a key to an address continues unabated.

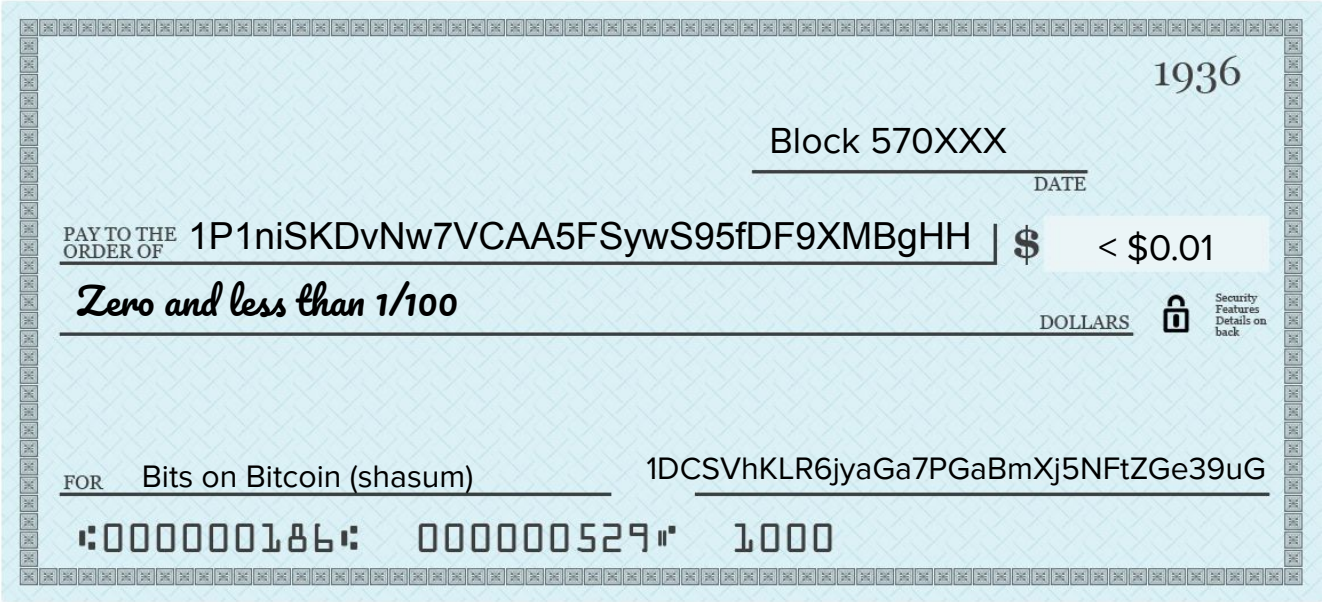


*In a standard base conversion, the 0x00 byte on the left would be irrelevant (like writing '052' instead of just '52'), but in the BTC network the left-most zero chars are carried through the conversion. So for every 0x00 byte on the left end of the binary address, we will attach one '1' character to the Base58 address. This is why main-network addresses all start with '1'

What I'm going to do:

CHECKSUM encode as 1ADDR

f178ed7f17154b292f7bc06aba2e35dcfb467d39 > 1P1niSKDvNw7VCAA5FSywS95fDF9XMBgHH

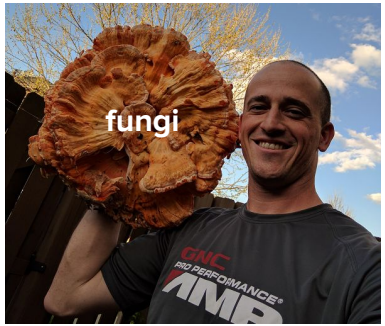


Demo

Justin P. Moore

B.S. Computer Science and Physics

M.S. Astrophysics, Ph.D. Candidate



demo — -bash — 114×28

```
-----------------------------------------------------------------
Step 1: Send a 550sat + enough to cover fee to developer address
Preferrably multiple UTXOs (split a large amount into lots of
fixed size UTXOs.
-----------------------------------------------------------------
1DCSVhKLR6jyaGa7PGaBmXj5NFtZGe39uG
-----------------------------------------------------------------
Step 2: Checking this address should now show a few UTXOs
-----------------------------------------------------------------
Step 3: Run shasum on your file
-----------------------------------------------------------------
shasum: 256:
Step 4: Compute the address that will encode this shasum:

f178ed7f17154b292f7bc06aba2e35dcfb467d39
-----------------------------------------------------------------
Step 5: Use/spend a UTXO to send 550satoshis.
-----------------------------------------------------------------
1DCSVhKLR6jyaGa7PGaBmXj5NFtZGe39uG
-----------------------------------------------------------------
Step 6: Generate raw tx from ^ to v
-----------------------------------------------------------------
1P1niSKDvNw7VCAA5FSywS95fDF9XMBgHH <-- address from step 4
-----------------------------------------------------------------
Step 7: Broadcast the tx to the network
-----------------------------------------------------------------
Include a link to the transaction on any block explorer.
Justins-MacBook-Pro:demo jmo$
```

Let’s talk about checksums.

Let’s talk about digital signatures

Step 1:
Create a specific Bitcoin address and load with UTXOs

How about using

1DCSVhKLR6jyaGa7PGaBmXj5NFtZGe39uG

which I mined.

DCSV - Decentralized Consensus Silicon Valley

Step 2:

Developer computes shasum and then pays small fee* to write data to the chain.

*small fee = 550 satoshis ($0.03 on BTC) less on other forks

MultiSig

Given most dev teams are n > 1 you could imagine releases being multisig transactions requiring m-of-n signatures before becoming official.

Depending on details of ASUS firmware backdoor, this could potentially have prevented it. (If they were using 1-of-1 signing keys.)

Incorporate this into the installation process

What if the application could check its own signature against what’s at a specific location on the chain such that it won’t continue installing without that information.

Think of it as an self-integrity check before installing.

Something like this could also potentially be used for software licensing and key management.

Checksum Example

Via Direct Download

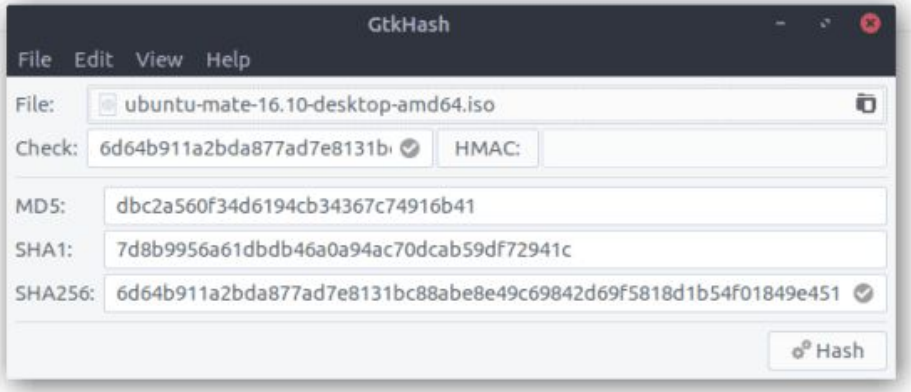
If preferred, you can also download the images over HTTP.

ubuntu-mate-16.10-desktop-amd64.iso

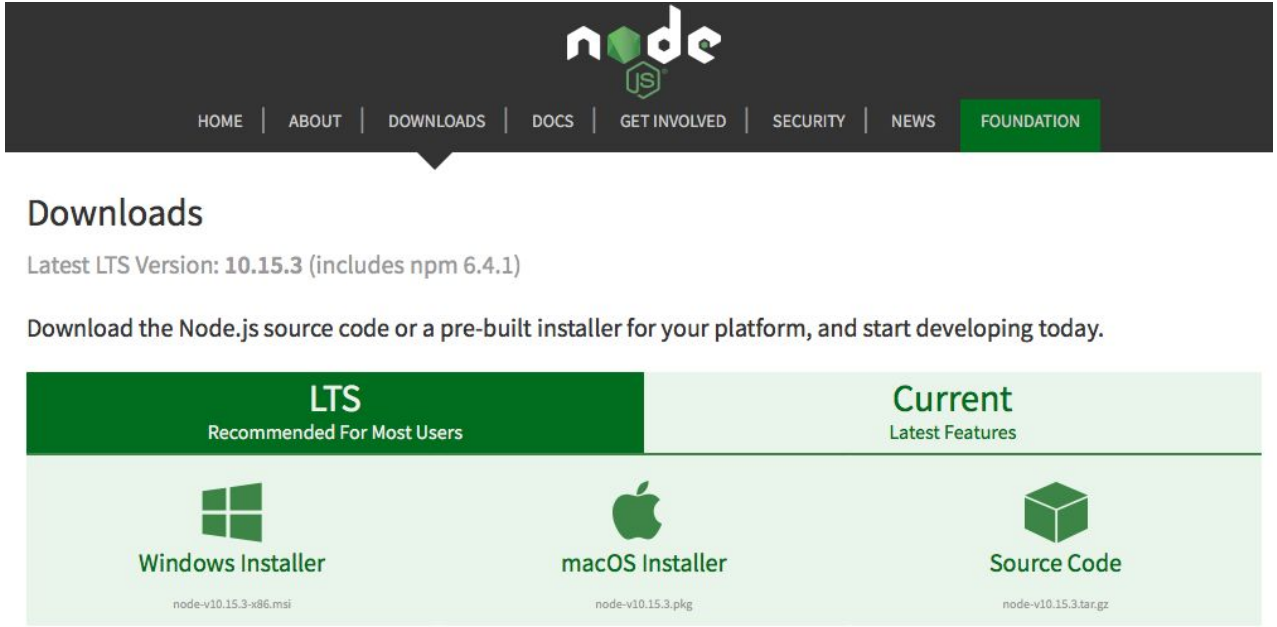
Download Size: 1.7 GB

SHA256 Checksum: 6d64b911a2bda877ad7e8131bc88abe8e49c69842d69f5818d1b54f01849e451

How to verify downloads



NodeJS (Checksum + Signature)



NodeJS

Additional Platforms

SmartOS Binaries	64-bit
Docker Image	Official Node.js Docker Image
Linux on Power Systems	64-bit
Linux on System z	64-bit
AIX on Power Systems	64-bit

- Signed SHASUMS for release files (How to verify)

NodeJS


```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA256

f2f018418b6bfa263ec981f04f3fa5337724edae8d77fc3951cd36667ee720ea  node-v10.15.3-aix-ppc64.tar.gz
7a5eaa1f69614375a695ccb62017248e5dcc15b0b8edffa7db5b52997cf992ba  node-v10.15.3-darwin-x64.tar.gz
8e3df823a58c7b2de327540a0b57a9bcf3f706108fe65c4cde9a073caae68cee  node-v10.15.3-darwin-x64.tar.xz
9e97ee69072836bfbf2a85c4af627ed152574c30c5a32e40fbfcdfda8d9b562e  node-v10.15.3-headers.tar.gz
f690b8808ccfeb5959436073717b648e4bdc521e3217ab7092d5c033326f6133  node-v10.15.3-headers.tar.xz
c82cd99e01f6e26830f0b3e0465f12f92957ebd69a68c91c03228c2669104359  node-v10.15.3-linux-arm64.tar.gz
3d7abbf64bffb07c55168ca0f1c17be12b0d93affe9b6cadd39724649215fab9  node-v10.15.3-linux-arm64.tar.xz
72529b6f77d95f9422f6d1c6b88c1f921b00e5500a1c3ea05927f1ae3704133d  node-v10.15.3-linux-armv6l.tar.gz
94432c2944fc78c2d5e82103f73596a060451330839562c04c414067007c5997  node-v10.15.3-linux-armv6l.tar.xz
6958551264884cd479f15ed8d40673655a283ed3bd8552d04e8531cd3ccdf483  node-v10.15.3-linux-armv7l.tar.gz
af2106b08f68e0884caa505ea7e695facc5b4cd356f1e08258899e94cc4c5df0  node-v10.15.3-linux-armv7l.tar.xz
0544b08467384ba3b3a462d8907d12cea71ac371f3d118057905dd845be43aad  node-v10.15.3-linux-ppc64le.tar.gz
a2fcc2e1827d7a034f39aad8225b4dd72376ad19f7a7884645a512aeeedf4ab5  node-v10.15.3-linux-ppc64le.tar.xz
073e6e2ad4e3a7580d87e5b70b9c1ce785b15e849dfd4f2f846c3039ad1e116c  node-v10.15.3-linux-s390x.tar.gz
545caa31bf06b150861ca3a2b1f5112aa92bb855de20fd98f8b7bc3f4c4311d7  node-v10.15.3-linux-s390x.tar.xz
6c35b85a7cd4188ab7578354277b2b2ca43eacc864a2a16b3669753ec2369d52  node-v10.15.3-linux-x64.tar.gz
faddbe418064baf2226c2fcbd038c3ef4ae6f936eb952a1138c7ff8cfe862438  node-v10.15.3-linux-x64.tar.xz
f4d0b944618afae2835b500e0cc1c5a013912597fce5560cd4bcb534f5270754  node-v10.15.3.pkg
c678b8e5a2d652f920c1093e6249b08e4746c2d37a5b9f719d04f3243776fb01  node-v10.15.3-sunos-x64.tar.gz
3732ae66ad564c192ff3a4a6e66e0d8922823c128bb8a6766ece87226982ad54  node-v10.15.3-sunos-x64.tar.xz
db460a63d057ac015b75bb6a879fcbe2fefaaf22afa4b6f6445b9db61ce2270d  node-v10.15.3.tar.gz
4e22d926f05415000205547 4e452ed6cbb85860aa7dc5422213a2002ed9791d5  node-v10.15.3.tar.xz
9df98cac063229aca443c040fd342a96667891bb8eda821d10aa4d49347d7add  node-v10.15.3-win-x64.7z
93c881fdc0455a932dd5b506a7a03df27d9fe36155c1d3f351ebfa4e20bf1c0d  node-v10.15.3-win-x64.zip
597a372964252daaba4cb8dcac57305f79cffeeca579625f0cd6ab85d29ccdda  node-v10.15.3-win-x86.7z
fc28bbd08b3d9b621c7c0ecd2b42506ca2f356f31f2b64210f413b34cff31799  node-v10.15.3-win-x86.zip
46b3d03c96de0b9e7d3a204c67772759283221f5e58ac225df813076a65e2738  node-v10.15.3-x64.msi
e73398cde3e054da7a0a05a86aa512a47a24b961b0659be30a0f01606ca234a9  node-v10.15.3-x86.msi
a921d1a4fa463e877087b3f25abd0ab05b63489bffcc9ff47acbbeee4e1b7494  win-x64/node.exe
4ed045ae1ba046506948b8f90c02716178cb0084f3b56866ac8d23b591e83235  win-x64/node.lib
538c8cc4e0b93facb9d63ed6c55d765ec33a18dd264c6c8b9415ad242521d8e6  win-x64/node_pdb.7z
525ea4adfd5c166076b273db6c0803283c57c4116fce56229ce87c8eb9fcdd25  win-x64/node_pdb.zip
39efb2a884d2f73680b986534eed000017ce16993ea9d695351593ffb9a7bb34  win-x86/node.exe
efed715422fcb7032290ec3c7e3b324126e082ee3a87d6ac497f6c97549e478e  win-x86/node.lib
38775185b6f6c090e7039ea0b3e630f4ab83e5c259d8d94f0f35f04ec12c0e98  win-x86/node_pdb.7z
1848e05e130dda3c3b53830cb78c4b28c137c7aac0890b70a8c863798c332ed5  win-x86/node_pdb.zip
-----BEGIN PGP SIGNATURE-----

iQEzBAEBCAAdFiEETtd49TnjY0x3nIfGlwYoSKGrAFwFAlx+rpcACgkQlwYoSKGr
AFzn6Qf/eLpfFexv+5ahnPq8ChYib9GKqNejkewhx9f/DOZ6vSj2NOd35bjoKrCk
/plFRWIdfsZ8ZqhVd4LQeWBxYVqdyRTM7zA6oLuNZwCxjuK0vwgnQZoq+LLnEal5
bsLAwdJJs3mEwvhNhZnQUjt1XFQDR+cBosuEArSoUlqSqUzfwo53x4eyJD5NcE77
944xFi7uob828J2wMebM1L5w0MHIRgqs9ptDHEigERGW4JFakkXHpsT9gT9IgFzx
l0FQH5oD5z7dZEWo19GfOVzsSr0kGf89VVX6y+b/nS8q4mG6x5LZKL+CecEofCr7
pVilEZXVzWQnVbdsTPXHNF/o0AHuCw==
=X9UW
-----END PGP SIGNATURE-----
```

NodeJS

Source Code	node-v10.15.3.tar.gz

Additional Platforms

SmartOS Binaries	64-bit
Docker Image	Official Node.js Docker Image
Linux on Power Systems	64-bit
Linux on System z	64-bit
AIX on Power Systems	64-bit

- Signed SHASUMS for release files (How to verify)

NodeJS

Verifying Binaries

Download directories contain a `SHASUMS256.txt` file with SHA checksums for the files.

To download `SHASUMS256.txt` using `curl`:

```
$ curl -O https://nodejs.org/dist/vx.y.z/SHASUMS256.txt
```

To check that a downloaded file matches the checksum, run it through `sha256sum` with a command such as:

```
$ grep node-vx.y.z.tar.gz SHASUMS256.txt | sha256sum -c -
```

For Current and LTS, the GPG detached signature of `SHASUMS256.txt` is in `SHASUMS256.txt.sig`. You can use it with `gpg` to verify the integrity of `SHASUM256.txt`. You will first need to import the GPG keys of individuals authorized to create releases. To import the keys:

```
$ gpg --keyserver pool.sks-keyservers.net --recv-keys DD8F2338BAE7501E3DD5AC78C273792F7D83545D
```

See the bottom of this README for a full script to import active release keys.

Next, download the `SHASUMS256.txt.sig` for the release:

```
$ curl -O https://nodejs.org/dist/vx.y.z/SHASUMS256.txt.sig
```

Then use `gpg --verify SHASUMS256.txt.sig SHASUMS256.txt` to verify the file's signature.