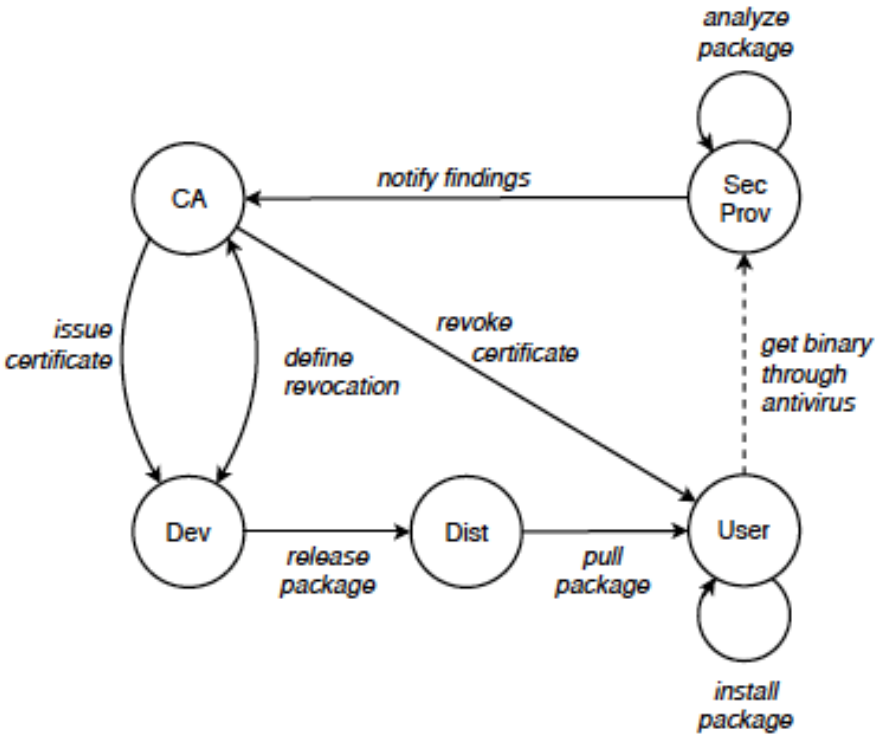

FIG. 1 (PRIOR ART)

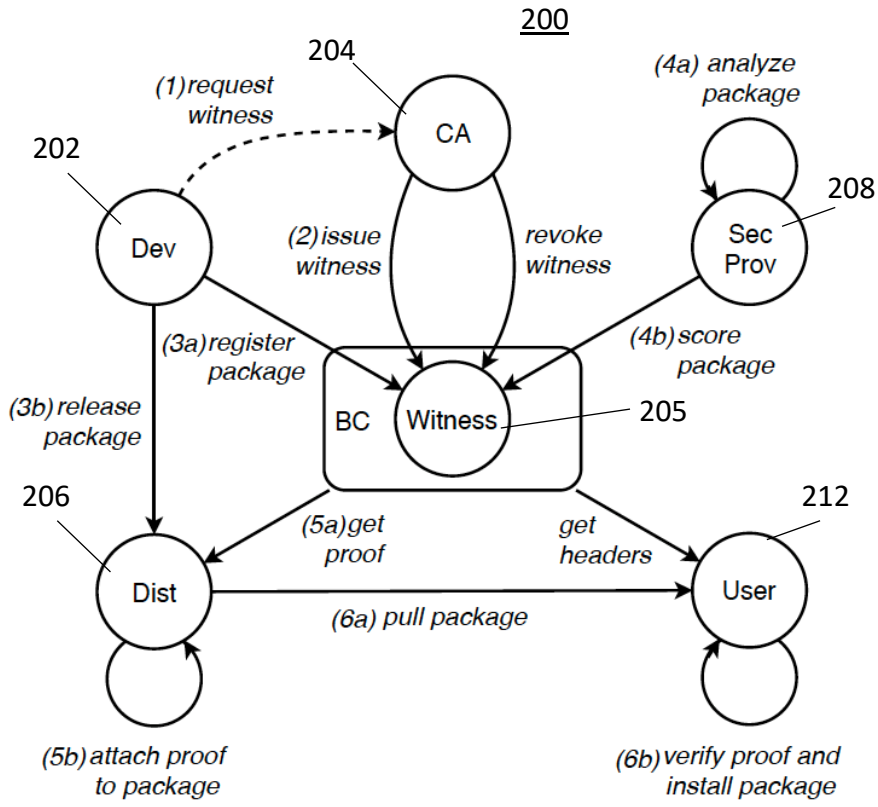

FIG. 2A

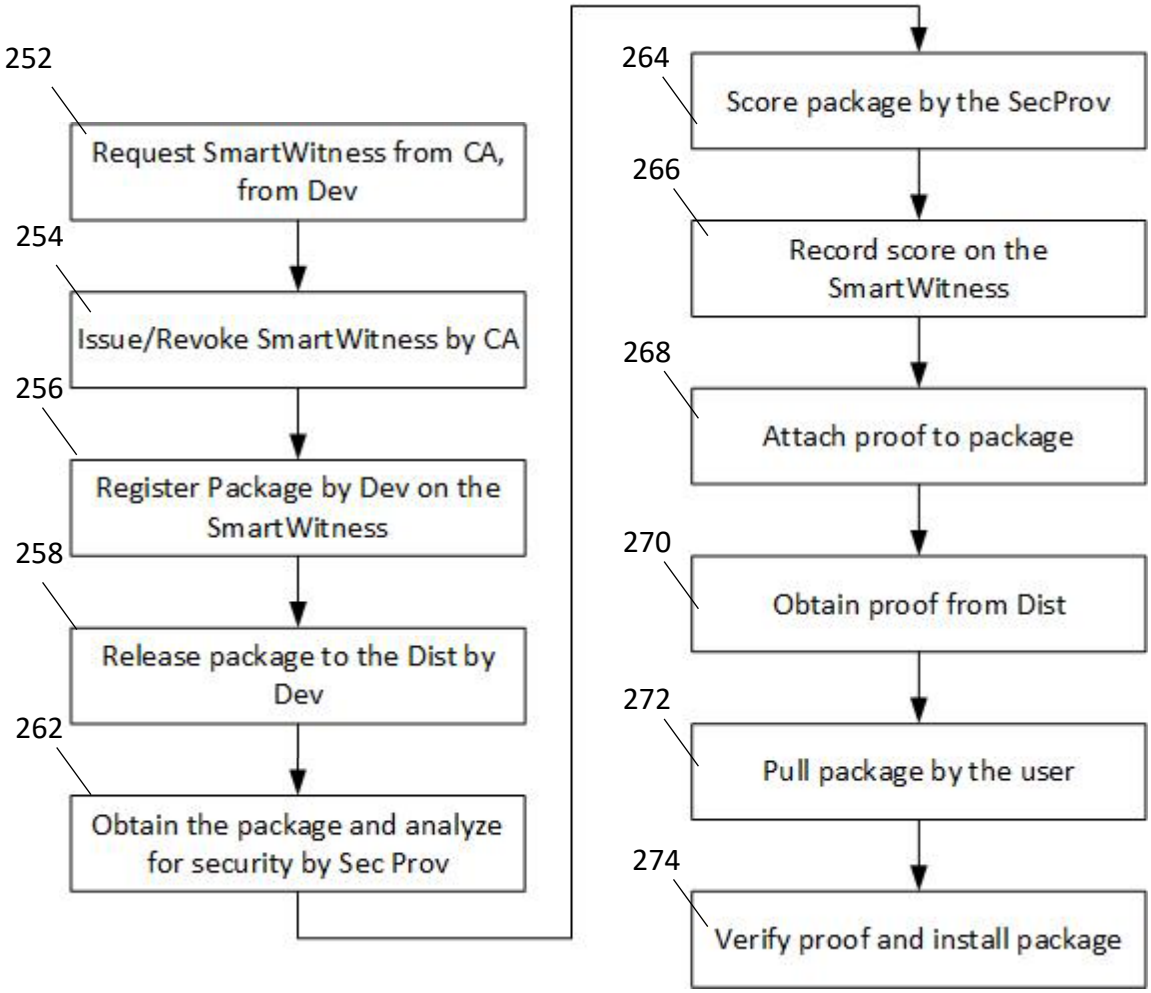
252
Request SmartWitness from CA, from Dev
254
Issue/Revoke SmartWitness by CA
256
Register Package by Dev on the SmartWitness
258
Release package to the Dist by Dev
262
Obtain the package and analyze for security by Sec Prov
264
Score package by the SecProv
266
Record score on the SmartWitness
268
Attach proof to package
270
Obtain proof from Dist
272
Pull package by the user
274
Verify proof and install package

FIG. 2B

Algorithm 1 CA Witness Factory Pseudo-Code

```
contract CA_{owner,providers}

    internal function initialize()
        owner ← msg.sender
        providers ← ∅
    end function

    function issueWitness(dev_addr)
        assert(msg.sender = owner)
        witness_addr ← Witness(dev_addr)
        if ¬witness_addr then
            return ⊥
        end if
        event WitnessCreated(dev_addr, witness_addr)
    end function

    function authorizeProvider(prov_addr)
        assert(msg.sender = owner)
        providers[prov_addr] ← true
        event ProviderRegistered(prov_addr)
    end function

    function unauthorizeProvider(prov_addr)
        assert(msg.sender = owner)
        providers[prov_addr] ← false
        event ProviderDeactivated(prov_addr)
    end function

    function validateProvider(prov_addr)
        if prov_addr ∉ providers then
            return ⊥
        end if
        return providers[prov_addr]
    end function

    function revokeWitness(addr)
        assert(msg.sender = owner)
        Witness(addr).revoke()
        event WitnessRevoked(addr)
    end function

    function renewWitness(addr, renewal_blocks)
        assert(msg.sender = owner)
        Witness(addr).renew(renewal_blocks)
        event WitnessRenewed(addr, renewal_blocks)
    end function
end contract
```

FIG. 3

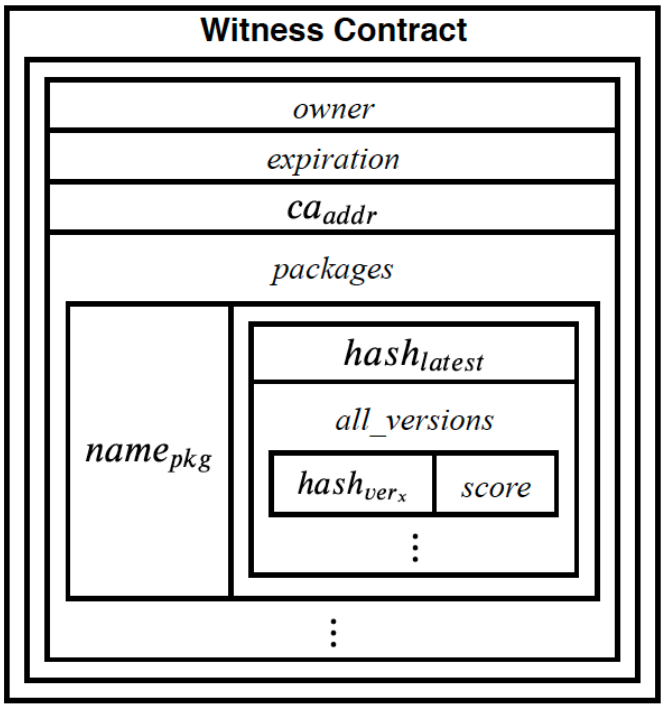

FIG. 4

Algorithm 2 SmartWitnesses Skeleton

contract Witness$_{owner,expiration,ca_{addr},packages}$

struct Package$_{latest,all_versions}$

internal function initialize(dev_{addr})
$owner \leftarrow dev_{addr}$
$ca_{addr} \leftarrow msg.sender$
$packages \leftarrow \emptyset$
end function

function registerPackage($name, hash_{bin}, version$)
assert($msg.sender = owner$)
if $expiration \leq block.number$ **then**
return $\perp$
end if
if $name \notin packages$ **then**
$packages[name] \leftarrow Package()$
end if
$pkg \leftarrow packages[name]$
if $hash_{bin} \in pkg.all_versions$ **then**
return $\perp$
end if
$pkg.all_versions[hash_{bin}] \leftarrow 0$
$pkg.latest = hash_{bin}$
event PkgRegistered($name, hash_{bin}, version$)
end function

function renew($renewal_{blocks}$)
assert($msg.sender = ca_{addr}$)
$expiration \leftarrow block.number + renewal_{blocks}$
end function

function revoke()
assert($msg.sender = ca_{addr}$)
$expiration \leftarrow block.number$
end function

function ratePackage($name, hash_{bin}, score, findings$)
$intf \leftarrow CA(ca_{addr})$
$prov \leftarrow intf.validateProvider(msg.sender)$
if $\neg prov \vee name \notin packages$ **then**
return $\perp$
end if
$pkg \leftarrow packages[name]$
if $hash_{bin} \notin pkg.all_versions$ **then**
return $\perp$
end if
$r_{new} \leftarrow scoreManagement(score, pkg, hash_{bin})$
$pkg.all_versions[hash_{bin}] \leftarrow r_{new}$
event PkgScored($name, hash_{bin}, score, findings$)
end function

function scoreManagement($score, pkg$)
assert($msg.sender = owner$)
... e.g., average, majority vote, etc.
end function

interface CA
function validateProvider($addr$)
end interface
end contract

FIG. 5

```
Package: telegram-desktop
Architecture: amd64
Version: 1.2.17-1
Priority: optional
Section: net
Maintainer: Ubuntu Developers <ubuntu-devel@ubuntu.com>
Original-Maintainer: Nicholas Guriev <guriev-ns@ya.ru>
Installed-Size: 30280
Depends: qt5-image-formats-plugins (>= 5.5), ...
Recommends: fonts-open-sans, libappindicator3-1
Filename: amd64/telegram-desktop_1.2.17-1_amd64.deb
Size: 16873104
MD5sum: 7b59a3393898ac78b4f8428e17ba37e9
SHA1: 222dc52b2d1215d26d92bd1358c512b5f373bd07
SHA256: f396c76cd843e69717a2e1ea5ff227fa2c1269dc235fb4...
SHA512: d256d93e71b48a409f3391ec4f0780a36f4224f1423070...
Dev-PK: ba8c620d97218a45ebe94cd019d01ea9f2baa0835922b3...
CA: 0xee35211C4D9126D520bBfeaf3cFee5FE7B86F221
CA-Nonce: 1
...
```

FIG. 6A

```
{
    "block-number": "0x3a5",
    "witness-proof": [
        '0xf9013180a0ab8cdb808c8303bb61fb48e276217b...',
        "0xf8b180808080a052bae369e3e99d512b416830be...",
        "0xf869a02030837a029b0fb0ab3d964f243eb52824..."
    ],
    "witness-address": "0xdb933c92ccafde43b11d8693c...",
    "storage-root": "0xa61cfd80c0977f5e7802d6867a700...",
    "storage-proof": [{
            "witness-owner": "0x8e240f19f0104b35f52...",
            "proof": [
                "0xf90211a0564872281b4e1a343eb1328b...",
                "0xf90191a0bb160302a7be12bda0e865b1...",
                "0xf891a0cceeb1e965d409338d5fd0c969...",
                "0xf69f3decd9548b62a8d60345a988386f..."
            ]
        }, {
            "witness-expiration": "0x5ed964ae",
            "proof": [
                "0xf90211a0564872281b4e1a343eb1328b...",
                "0xf90191a0bb160302a7be12bda0e865b1...",
                "0xf891a0cceeb1e965d409338d5fd0c969...",
                "0xf69f3decd9548b62a8d60345a988386f..."
            ]
        }, {
            "security-score": "0x0109",
            "proof": [
                "0xf90211a0564872281b4e1a343eb1328b...",
                "0xf901f1a0135ccb61f2b31d57d8f4e0fc...",
                "0xf851808080808080a0682277c9929d59...",
                "0xe19f3e4931d1b15e4dd6c362089bd25d..."
            ]
        }, {
            "hash-last-version": "0x93ffe4f6ae35257f...",
            "proof": [
                "0xf90211a0564872281b4e1a343eb1328bf...",
                "0xf90191a0056473048e52c0f8a40e0f2de...",
                "0xf851808080808080a062c53dd92f7a726..."
            ]
        }
    ]
}
```


FIG. 6B