Bolt

Contents

Bolt release notes

New features, enhancements, known issues, and resolved issues for the Bolt 1.x release series.

Version 1.26.0 released July 2019.

- **Bolt versioning on page 5**Bolt follows semantic versioning guidelines.
- **New features on page 5**New features added to Bolt in the 1.x release series.
- **Resolved issues on page 13**Security and bug fixes in the Bolt 1.x release series.
- **Known issues on page 16**Known issues for the Bolt 1.x release series.
- **Deprecations and removals on page 17**A list of the features and functions deprecated or removed from Bolt 1.x.

Bolt versioning

Bolt follows semantic versioning guidelines.

This system uses an "x.y.z" pattern, where "x" is the number of a major release, "y" indicates a minor release that introduces new features but does not include breaking changes, and "z" reflects a bug fix release.

Bolt API

The 1.0 series of Bolt is stable and be free of breaking changes to its public API. In general this means that plans, inventory and config tested with one 1.y series release of Bolt continues to work when used with a newer 1.y series release. There is, however, new functionality added in the 1.0 series. You can expect the following types of changes:

- New keys in JSON format output.
- Differences in log output and human format output.
- New methods added to plan Data Type Objects.
- New options added to plan functions.
- Support for new revisions of the task spec and newer tasks.
- Versions of bundled modules are updated and might include breaking changes.

Bolt Releases

For the 1.y series of Bolt we do not plan to backport bug fixes to any previously released version. You are encouraged to upgrade frequently and use the latest release.

Parent topic: Bolt release notes on page 5

Related information

Semantic Versioning

New features

New features added to Bolt in the 1.x release series.

Options for PCP transport now configurable in bolt.yaml \(1.26.0\)

The `job-poll-timeout` and `job-poll-interview` options for the `PCP` transport are now configurable in bolt.yaml. \(BOLT-1425\)

Task plugin improvements \(1.26.0\)

The `task` plugin now enables you to run a task to discover targets or look up configuration information in the version 2 inventory file. \(BOLT-1408\)

Ability to see nodes in an inventory group \(1.26.0\)

You can now see what nodes a Bolt command acts on using the `bolt inventory show` subcommand. Pass a targeting option, such as `-n node1,node2`, `-n groupname`, `-q query`, `--rerun`, and other targeting options to specify which nodes to list. \(BOLT-1398\)

Support for an apply step \(1.26.0\)

Yaml plans now support applying Puppet resources with a `resources` step. \(BOLT-1222\)

Help text only lists options for a given command \(1.24.0\)

Help text now only shows options for the specified subcommand and action. Previously, all options were displayed in the help text, even if those options did not apply to the specified subcommand and action. \(BOLT-1342\)

Packages for Fedora 30 \(1.24.0\)

Bolt packages are now available for Fedora 30. \(BOLT-1302\)

Adds support for embedding eyaml data in the inventory \(1.24.0\)

This change adds a hiera-eyaml compatible pkcs7 plugin and support for embedding eyaml data in the inventory. \(BOLT-1270\)

Allow `$nodes` as positional arg for `run_plan()` \(1.24.0\)

This change allows the `run_plan()` function to be invoked with `$nodes` as the second positional argument, so that it can be used the same way `run_task()` is used. \(BOLT-1197\)

`catch_errors` function \(1.23.0\)

The new plan function, `catch_errors`, accepts a list of types of errors to catch and a block of code to run where, if it errors, the plan continues executing. \(BOLT-1316\)

Forge `baseurl` setting in `puppetfile` config \(1.23.0\)

The `puppetfile` config section now supports a Forge subsection that you can use to set an alternate Forge location from which to download modules. \(BOLT-1376\)

Proxy configuration \(1.22.0\)

You can now specify an HTTP proxy for `bolt puppetfile install` in `bolt.yaml`, for example:

```
puppetfile:
  proxy: https://proxy.example.com
```

\(BOLT-1327\)

Support for version 4 Terraform state files \(1.22.0\)

Target-lookups using the Terraform plugin are now compatible with the version 4 Terraform state files generated by Terraform version 0.12.x. \(BOLT-1341\)

Prompt for sensitive data from inventory v2 \(1.22.0\)

A new `prompt` plugin in inventory v2 allows setting configuration values via a prompt. \(BOLT-1269\)

Set custom exec commands for Docker transport \(1.21.0\)

New configuration options, `shell-command` and `tty`, for the Docker transport allow setting custom Docker exec commands.

Check existence and readability of files \(1.21.0\)

New functions, `file::exists` and `file::readable`, test whether a given file exists and is readable, respectively. \(BOLT-1338\)

Output a message \(1.21.0\)

The new `out::message()` function can be used to print a message to the user during a plan. \(BOLT-1325\)

Return a filtered `ResultSet` with a `ResultSet` \(1.21.0\)

A new `filter_set` function in the `ResultSet` data type filters a `ResultSet` with a lambda to return a `ResultSet` object. \(BOLT-1337\)

Improved error handling for unreadable private keys \(1.21.0\)

A more specific warning is now surfaced when an SSH private key can't be read from Bolt configuration. \(BOLT-1297\)

Look up PuppetDB facts in inventory v2 \(1.21.0\)

The PuppetDB plugin can now be used to look up configuration values from PuppetDB facts for the `name`, `uri`, and `config` inventory options for each target. \(BOLT-1264\)

Terraform plugin in inventory v2 \(1.20.0\)

A new plugin in inventory v2 loads Terraform state and map resource properties to target parameters. This plugin enables using a Terraform projectto dynamically determine the targets to use when running Bolt. \(BOLT-1265\)

Type info available in plans \(1.20.0\)

A new `to_data` method is available for plan result objects that provides a hash representation of the object. \(BOLT-1223\)

Improved logging for `apply` \(1.20.0\)

The Bolt apply command and the `apply` function from plans now show log messages for changes and failures that happened while applying Puppet code. \(BOLT-901\)

Convert YAML plans to Puppet plans \(1.19.0\)

You can now convert YAML plans to Puppet plans with the `bolt plan convert` command. \(BOLT-1195\)

Improved error handling for missing commands \(1.19.0\)

A clear error message is now shown when no object is specified on the command line, for example `bolt command run --nodes <NODE_NAME>`. \(BOLT-1243\)

Inventory file version 2 \(1.18.0\)

An updated version of the inventory file, version 2, is now available for experimentation and testing. In addition to several syntax changes, this version enables setting a human readable name for nodes and dynamically populating groups from PuppetDB queries. This version of the inventory file is still in development and might experience breaking changes in future releases. \(BOLT-1232\)

YAML plan validation \(1.18.0\)

YAML plan validation now alerts on syntax errors before plan execution. \(BOLT-1194\)

Rerun failed commands \(1.17.0\)

Bolt now stores information about the last failed run in a `.rerun.json` file in the Bolt project directory. You can use this record to target nodes for the next run using `--retry failure` instead of `--nodes`.

For repositories that contain a Bolt project directory, add `$boltdir/.last_failure.json` to `.gitignore` files.

Stored information may include passwords, so if you save passwords in URIs, set `save-failures: false` in your Bolt config file to avoid writing passwords to the `.rerun.json` file. \(BOLT-843\)

Packaged `hiera-eyaml` gem \(1.16.0\)

Bolt packages now include the `hiera-eyaml` gem. \(BOLT-1026\)

Local transport options for `run-as`, `run-as-command`, and `sudo-password` \(1.16.0\)

The local transport now accepts the `run-as`, `run-as-command`, and `sudo-password` options on non-Windows nodes. These options escalate the system user \(who ran `bolt`\) to the specified user, and behave like the same options using the SSH transport. `_run_as` can also be configured for individual plan function calls for the local transport. \(BOLT-1052\)

`Localhost` target applies the `puppet-agent` feature \(1.16.0\)

When the target hostname is `localhost`, the `puppet-agent` feature is automatically added to the target, because the Puppet agent installed with Bolt is present on the local system. This functionality is available on all transports, not just the local transport. \(BOLT-1200\)

Tasks use the Bolt Ruby interpreter only for `localhost` \(1.16.0\)

Bolt sets its own installed Ruby as the default interpreter for all `.rb` scripts running on `localhost`. Previously, this default was used on all commands run over the local transport; it's now used when the hostname is `localhost` regardless of the transport. \(BOLT-1205\)

Fact indicates whether Bolt is compiling a catalog \(1.16.0\)

If Bolt is compiling a catalog, `$facts['bolt']` is set to `true`, allowing you to determine whether modules are being used from a Bolt catalog. \(BOLT-1199\)

YAML plans \(1.15.0\)

You can now write plans in the YAML language. YAML plans run a list of steps in order, which allows you to define simple workflows. Steps can contain embedded Puppet code expressions to add logic where necessary. For more details about YAML plans, see Writing plans in YAML. For an example of a YAML plan in use, see the Puppet blog. \(BOLT-1150\)

This version also adds analytics data collection about the number of steps and the return type of YAML plans. \(BOLT-1193\)

Support for Red Hat Enterprise Linux 8 \(1.15.0\)

A Bolt package is now available for RHEL 8. \(BOLT-1204\)

Improved load time \(1.15.0\)

Bolt startup is now more efficient. \(BOLT-1119\)

Details about `Result` and `ResultSet` objects \(1.15.0\)

The `Result` and `ResultSet` objects now include information in the JSON output about the action that generated the result. \(BOLT-1125\)

Inventory warning about unexepected keys \(1.15.0\)

An informative warning message is now logged when invalid `group` or `node` configuration keys are detected in the `inventoryfile`. \(BOLT-1017\)

`BoltSpec::Run` support for uploading files to remote systems \(1.15.0\)

`BoltSpec::Run` now supports the `upload_file` action. \(BOLT-953\)

Support for Puppet device modules in a manifest block \(1.14.0\)

You can now apply Puppet code on targets that can't run a Puppet agent using the remote transport via a proxy. This is an experimental feature and might change in future minor \(y\) releases. \(BOLT-645\)

Validation and error handling for invalid PCP tokens \(1.14.0\)

The PCP transport `token-file` configuration option now includes validation and a more helpful error message. \(BOLT-1076\)

SMB file transfer on Windows \(1.13.0\)

When transferring files to a Windows host, you can now optionally use the SMB protocol to reduce transfer time. You must have either administrative rights to use an administrative share, like `\\host\C$`, or use UNC style paths to access existing shares, like `\\host\share`. You can use SMB file transfers only over HTTP, not HTTPS, and SMB3, which supports encryption, is not yet supported. \(BOLT-153\)

Interpreter configuration option \(1.13.0\)

An `interpreters` configuration option enables setting the interpreter that is used to execute a task based on file extension. This options lets you override the shebang defined in the task source code with the path to the executable on the remote system. \(BOLT-146\)

Improved error handling \(1.13.0\)

Clearer error messages now alert you when you use plan functions not meant to be called in apply blocks. \(BOLT-1131\)

Updated project directory structure \(1.12.0\)

Within your project directory, we now recommend using a directory called `site-modules`, instead of the more ambiguously named `site`, to contain any modules not intended to be managed with a Puppetfile. Both `site-modules` and `site` are included on the default modulepath to maintain backward compatibility. \(BOLT-1108\)

`bolt puppetfile show-modules` command \(1.12.0\)

A new `bolt puppetfile show-modules` command lists the modules, and their versions, installed in the current `Boltdir`. \(BOLT-1118\)

BoltSpec::Run helpers accept options consistently \(1.12.0\)

All BoltSpec::Run helpers now require the params or arguments argument to be passed. \(BOLT-1057\)

bolt task show displays module path \(1.11.0\)

Task and plan list output now includes the module path to help you better understand why a task or plan is not included. \(BOLT-1027\)

PowerShell scripts over the PCP transport \(1.11.0\)

You can now run PowerShell scripts on Windows targets over the PCP transport. \(BOLT-830\)

RSA keys with OpenSSH format \(1.11.0\)

RSA keys stored in the OpenSSH format can now be used for authentication with the SSH transport. \(BOLT-1124\)

Support for new platforms \(1.11.0\)

Bolt packages are now available for these platforms:

- Fedora 28 and 29 \(BOLT-978\)
- macOS 10.14 Mojave \(BOLT-1040\)

Hyphens allowed in aliases and group names \(1.10.0\)

Node aliases and group names in the Bolt inventory can now contain hyphens. \(BOLT-1022\)

Improved out-of-the-box tasks \(1.9.0\)

The package and service tasks now select task implementation based on available target features while their platform-specific implementations are private. \(BOLT-1049\)

Respect multiple PuppetDB server_urls \(1.9.0\)

Bolt now tries to connect to all configured PuppetDBserver_urls before failing. \(BOLT-938\)

Standard library functions \(1.8.0\)

Bolt now includes several standard library functions useful for writing plans, including:

- ctrl::sleep
- ctrl::do_until
- file::read
- file::write
- system::env

See Plan execution functions and standard libraries for details. \(BOLT-1054\)

Configure proxy SSH connections through jump hosts \(1.7.0\)

You can now configure proxy SSH connections through jump hosts from the inventory file with the proxyjump SSH configuration option. \(BOLT-1039\)

Query resource states from a plan \(1.7.0\)

You can now query resource states from a plan with the get_resources function. \(BOLT-1035\)

Specify an array of directories in `modulepath` \(1.7.0\)

You can now specify an array of directories for the `modulepath` setting in `bolt.yaml`, rather than just a string. This change enables using a single `bolt.yaml` on both *nix and Windows clients. \(BOLT-817\)

Save keystrokes on `modulepath`, `inventoryfile`, and `verbose` \(1.7.0\)

You can now use shortened command options for `modulepath` \(`-m`\), `inventoryfile` \(`-i`\), and `verbose` \(`-v`\). \(BOLT-1047\)

Remote tasks \(1.6.0\)

You can now run tasks on a proxy target that remotely interacts with the real target, as defined by the `run-on` option. Remote tasks are useful for targets like network devices that have limited shell environments, or cloud services driven only by HTTP APIs. Connection information for non-server targets, like HTTP endpoints, can be stored in inventory. \(BOLT-791\)

`reboot` module plan \(1.6.0\)

Bolt now ships with the `reboot` module, and that module now provides a plan that reboots targets and waits for them to become available. \(BOLT-459\)

Local transport on Windows \(1.6.0\)

The `local` transport option is now supported on Windows. \(BOLT-608\)

`bolt_shim` module contents marked as sensitive \(1.6.0\)

The `bolt_shim` module that enables using Bolt with PE now marks file content as sensitive, preventing it from being logged or stored in a database. \(BOLT-815\)

Node aliases \(1.5.0\)

You can now specify aliases for nodes in your inventory and then use the aliases to refer to specific nodes. \(BOLT-510\)

Run `apply` with PE orchestrator without installing `puppet_agent` module \(1.5.0\)

Bolt no longer requires installing the `puppet_agent` module in PE in order to run `apply` actions with the PE orchestrator. \(BOLT-940\)

Bolt `apply` with orchestrator \(1.4.0\)

A new `puppetlabs-apply_helper` module enables using Bolt `apply` with orchestrator. For details, see the module README. \(BOLT-941\)

Add targets to a group \(1.4.0\)

A new `add_to_group` function allows you to add targets to an inventory group during plan execution. \(BOLT-942\)

Additional plan test helpers \(1.4.0\)

The `BoltSpec::Plans` library now supports unit testing plans that use the `_run_as` parameter, `apply`, `run_command`, `run_script`, and `upload_file`. \(BOLT-984\)

Data collection about applied catalogs \(1.4.0\)

If analytics data collection is enabled, we now collect randomized info about the number of statements in an apply block, and how many resources that produces for each target. \(BOLT-644\)

Docker transport for running commands on containers \(1.3.0\)

A new Docker transport option enables running commands on container instances with the Docker API. The Docker transport is experimental because the capabilities and role of the Docker API might change.\(BOLT-962\)

Wait until all target nodes accept connections \(1.3.0\)

A new `wait_until_available` function waits until all targets are accepting connections, or triggers an error if the command times out. \(BOLT-956\)

Apply Puppet manifest code with bolt apply command \(1.2.0\)

The command `bolt apply` has been added to apply Puppet manifest code on targets without wrapping them in an `apply()` block in a plan.

Note: This command is in development and subject to change.

\(BOLT-858\)

Python and Ruby helper libraries for tasks \(1.2.0\)

Two new libraries have been added to help you write tasks in Ruby and Python:

- https://github.com/puppetlabs/puppetlabs-ruby_task_helper
- https://github.com/puppetlabs/puppetlabs-python_task_helper

Use these libraries to parse task input, catch errors, and produce task output. For details, see Task Helpers. \(BOLT-906 and BOLT-907\)

Redacted passwords for printed target objects \(1.2.0\)

When the `Target` object in a Bolt plan is printed, it includes only the host, user, port, and protocol used. The values for `password` and `sudo-password` are redacted. \(BOLT-944\)

Share code between tasks \(1.1.0\)

Bolt includes the ability to share code between tasks. A task can include a list of files that it requires, from any module, that it copies over and makes available via a _installdir parameter. This feature is also supported in Puppet Enterprise 2019.0. For more information see, Sharing task code. \(BOLT-755\)

Upgraded WinRM gem dependencies \(1.1.0\)

The following gem dependencies have been upgraded to fix the connection between OMI server on Linux and the WinRM transport:

- winrm 2.3.0
- winrm-fs 1.3.1
- json-schema 2.8.1

\(BOLT-929\)

Mark internal tasks as private \(1.1.0\)

In the task metadata, you can mark internal tasks as private and prevent them from appearing in task list UIs. \(BOLT-734\)

Upload directories via plans \(1.1.0\)

The `bolt file upload` command and `upload_file` action now upload directories. For use over the PCP transport these commands require puppetlabs-bolt_shim 0.2.0 or later. \(BOLT-191\)

Support for public-key signature system ed25519 \(1.1.0\)

The ed25519 key type is now supported out-of-the-box in Bolt packages. \(BOLT-380\)

Parent topic: Bolt release notes on page 5

Resolved issues

Security and bug fixes in the Bolt 1.x release series.

Modulepath now handles folder names in uppercase characters on Windows \(1.26.0\)

Bolt now prints a warning stating that it is case sensitive when the specified path is not found but another path is found with different capitalization. For example, if the actual path is `C:\User\Administrator\modules` but the user specifies `C:\user\administrator\modules`, a warning states that the specified path was not used and that the correct path is `C:\User\Administrator\modules`. \(BOLT-1318\)

`out::message` didn't work inside `without_default_logging` \(1.25.0\)

The `out::message` standard library plan function now works within a `without_default_logging` block. \(BOLT-1406\)

Task action stub parameter method incorrectly merged options and arguments \(1.25.0\)

When a task action stub expectation fails, the expected parameters are now properly displayed. \(BOLT-1399\)

The `wait_until_available` function returned incorrect results using orchestrator \(1.23.0\)

When using the PCP transport, the plan function `wait_until_available` now returns error results only for targets that can't be reached. \(BOLT-1382\)

PowerShell tasks on localhost didn't use correct default PS_ARGS \(1.23.0\)

PowerShell scripts and tasks run over the local transport on Windows hosts no longer load profiles and are run with the `Bypass` execution policy to maintain parity with the WinRM transport. \(BOLT-1358\)

Inventory was loaded for commands that didn't use it \(1.20.0\)

Inventory was loaded even for commands that don't use targets, such as `bolt task show`. An error in the inventory could subsequently cause the command to fail. \(BOLT-1268\)

YAML plan converter wrapped single-line evaluation steps \(1.20.0\)

The `bolt plan convert` command wrapped single-line evaluation steps in a `with()` statement unnecessarily. \(BOLT-1299\)

File upload stalled with local transport using `run-as` \(1.18.0\)

The `bolt file upload` command stalled when using local the local transport if the destination file existed. \(BOLT-1262\)

Rerun file wasn't generated without an existing project directory \(1.18.0\)

If no Bolt project directory existed, a `.rerun.json` file wasn't created, preventing you from rerunning failed commands. Bolt now creates a default project directory when one doesn't exist so that `.rerun.json` files are generated as expected. \(BOLT-1263\)

SELinux management didn't work on localhost \(1.17.0\)

Bolt now ships with components similar to the Puppet agent to avoid discrepancies between using a puppet-agent to apply Puppet code locally versus using the Bolt puppet-agent. \(BOLT-1244\)

Linux implementation of the service and package tasks returned incorrect results \(1.16.0\)

The PowerShell and bash implementations for the service and package tasks are more robust and provide output more consistent with the Ruby implementation. \(BOLT-1103, BOLT-1104\)

Remote tasks could run on non-remote targets \(1.15.0\)

Remote tasks can now be run only on remote targets \(BOLT-1203\)

`known_hosts` weren't parsed correctly \(1.15.0\)

Previously, when a valid hostname entry was present in `known_hosts` and the `host-key-check` SSH configuration option was set, host key validation could fail when a valid IP address was not included in the `known_hosts` entry. This behavior was inconsistent with system SSH where the IP address is not required. Host key checking has been updated to match system SSH. \(BOLT-495\)

Plan variables were visible to sub-plans \(1.15.0\)

Variables defined in scope in a plan were visible to sub-plans called with `run_plan`. \(BOLT-1190\)

The `_run_as` option was clobbered by configuration \(1.13.1\)

The `run-as` configuration option took precedence over the `_run_as` parameter when calling `run_*` functions in a plan. The `_run_as` parameter now has a higher priority than config or CLI. \(BOLT-1050\)

Tasks with certain configuration options failed when using `stdin` \(1.13.1\)

When both `interpreters` and `run-as` were configured, tasks that required parameters to be passed over `stdin` failed. \(BOLT-1155\)

Ruby task helper symbolized only top-level parameter keys \(1.13.0\)

Previously the `ruby_task_helper``TaskHelper.run` method symbolized only-top level parameter keys. Now nested keys are also symbolized. \(BOLT-1053\)

String segments in commands had to be triple-quoted in PowerShell \(1.12.0\)

When running Bolt in PowerShell with commands to be run on *nix nodes, string segments that could be interpreted by PowerShell needed to be triple-quoted. \(BOLT-159\)

Unsecured download of the `puppet_agent::install` task \(1.11.0\)

The bash implementation of the `puppet_agent::install` task now downloads packages over HTTPS instead of HTTP. This fix ensures the download is authenticated and secures against a man-in-the-middle attack.

Unsecured download of the `puppet_agent::install_powershell` task \(1.10.0\)

The PowerShell implementation of the `puppet_agent::install` task now downloads Windows .msi files using HTTPS instead of HTTP. This fix ensures the download is authenticated and secures against a man-in-the-middle attack.

Bolt crashed if PuppetDB configuration was invalid \(1.9.0\)

If an invalid `puppetdb.conf` file is detected, Bolt now issues a warning instead of crashing \(BOLT-756\)

Local transport returned incorrect exit status \(1.9.0\)

Local transport now correctly returns an exit code instead of the stat of the process status as an integer. \(BOLT-1074\)

Standard library functions weren't packaged in 1.8.0 \(1.8.1\)

Version 1.8.0 didn't include new standard library functions as intended. This release now includes standard library functions in the gem and packages. \(BOLT-1065\)

`puppet_agent::install` task didn't match on Red Hat \(1.8.0\)

The `puppet_agent::install` task now uses updates in the `facts` task to resolve Red Hat operating system facts and to download the correct `puppet-agent` package. \(BOLT-997\)

Select module content missing from `puppet-bolt` package \(1.7.0\)

Previous releases of the `puppet-bolt` package omitted the `python_task_helper` and `ruby_task_helper` modules. These are now included. \(BOLT-1036\)

`wait_until_available` function didn't work with Docker transport \(1.6.0\)

We merged the Docker transport and `wait_until_available` function in the same release, and they didn't play nicely together. \(BOLT-1018\)

Python task helper didn't generate appropriate errors \(1.6.0\)

The Python task helper included with Bolt didn't produce an error if an exception was thrown in a task implemented with the helper. \(BOLT-1021\)

Plans with no return value weren't marked complete in PE \(1.3.0\)

Bolt now correctly reports plan completion to PE for plans that don't return a value. Previously, a plan that didn't return a value incorrectly logged that the plan didn't complete. \(BOLT-959\)

Some functions weren't available in the BoltSpec::Plans library \(1.3.0\)

The BoltSpec::Plans library now supports plans that use `without_default_logging` and `wait_until_available`, and includes a setup helper that ensures tasks are found and that `notice` works. \(BOLT-971\)

Task implementation not located relative to other files in installdir \(1.2.0\)

When you use tasks that include shared code, the task executable is located alongside shared code at `_installdir/MODULE/tasks/TASK`. \(BOLT-931\)

Error when puppet_agent task not run as root \(1.1.0\)

The puppet_agent task now checks that it is run as root. When run as another user, it prints and fails with a helpful message. \(BOLT-878\)

Bolt suppresses errors from transport \(1.1.0\)

Previously, Bolt suppressed some exception errors thrown by transports. For example, when the ed25519 gem was not present for an Net::SSH process, the NotImplementedError for ed25519 keys would not appear. These errors are now identified and displayed. \(BOLT-922\)

Loading bolt/executor is "breaking" gettext setup in spec tests \(1.0.0\)

When Bolt is used as a library, it no longer loads code from r10k unless you explicitly `require 'bolt/cli'`. \(BOLT-914\)

Deprecated functions in stdlib result in Evaluation Error \(1.0.0\)

Manifest blocks will now allow use of deprecated functions from stdlib, and language features governed by the 'strict' setting in Puppet. \(BOLT-900\)

Bolt apply does not provide clientcert fact \(1.0.0\)

`apply_prep` has been updated to collect agent facts as listed in Puppet agent facts. \(BOLT-898\)

C:\\Program Files\\Puppet Labs\\Bolt\\bin\\bolt.bat is non-functional \(1.0.0\)

When moving to Ruby 2.5, the .bat scripts in Bolt packaging reverted to hard-coded paths that were not accurate. As a result Bolt would be unusable outside of PowerShell. The .bat scripts have been fixed so they work from cmd.exe as well. \(BOLT-886\)

Parent topic: Bolt release notes on page 5

Known issues

Known issues for the Bolt 1.x release series.

JSON strings as command arguments might require additional escaping in PowerShell

When passing complex arguments to tasks with `--params`, JSON strings \(typically created with the `ConvertTo-Json` cmdlet\) might require additional escaping. In some cases, you can use the PowerShell stop parsing symbol `--%` as a workaround. \(BOLT-1130\)

SSH keys generated with ssh-keygen from OpenSSH 7.8+ fail

OpenSSH 7.8 switched to generating private keys with its own format rather than the OpenSSL PEM format. The Bolt SSH implementation assumes any key using the OpenSSH format uses ed25519, resulting in false errors such as:

```
 OpenSSH keys only supported if ED25519 is available net-ssh requires
 the following gems for ed25519 support: * ed25519 (>= 1.2, < 2.0) *
 bcrypt_pbkdf (>= 1.0, < 2.0) See https://github.com/net-ssh/net-ssh/
issues/565 for more information Gem::LoadError : "ed25519 is not part of the
 bundle. Add it to your Gemfile."
```

or

```
Failed to connect to HOST: expected 64-byte String, got NUM
```

As a workaround, you can generate new keys with the ssh-keygen `-m PEM` flag. For existing keys, you can try exporting keys from the OpenSSH format using the `-e` option, although export is not implemented for all private key types. \(BOLT-920\)

Commands fail in remote Windows sessions

Interactive tools fail when run in a remote PowerShell session. For example, using `--password` to prompt for a password when running Bolt triggers an error. As a workaround, consider putting the password in `bolt.yaml` or an inventory file, or passing the password on the command line. \(BOLT-1075\)

No Kerberos support

A license incompatibility with other components distributed with Bolt prevents authenticating with Kerberos over SSH using the net-ssh-krb gem. \(BOLT-980\)

Kerberos over WinRM, both from Windows and non- Windows hosts, is also not supported. \(BOLT-126\)

Parent topic: Bolt release notes on page 5

Deprecations and removals

A list of the features and functions deprecated or removed from Bolt 1.x.

`lookups` removed from `target_lookups` \(0.25.0\)

We have deprecated the `target-lookups` key in the experimental inventory file v2. To address this change, migrate any `target-lookups` entries to `targets` and move the `plugin` key in each entry to `_plugin`.

Configuration location ~/.puppetlab/bolt.yaml \(0.21.0\)

When the directory Boltdir was added as the local default configuration directory, the previous directory, `~/.puppetlab/bolt.yaml`, was deprecated in favor of `~/.puppetlabs/bolt/bolt.yaml`. For more information on the current default directory for configfile, inventoryfile and modules, see Configuring Bolt. \(BOLT-503\)

Parent topic: Bolt release notes on page 5