



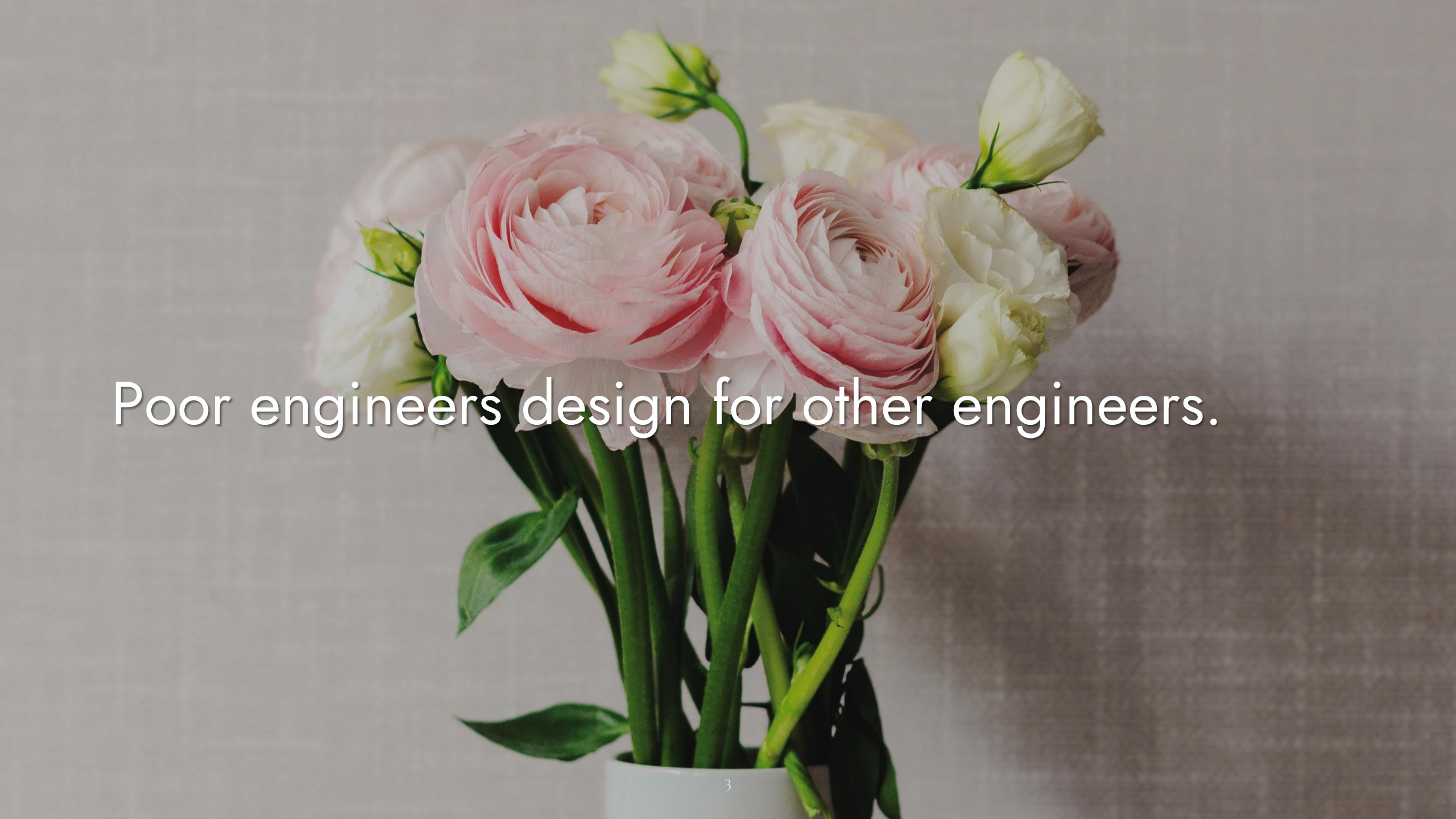
PRACTICAL MAGIC: THE RESILIENCE POTION AND SECURITY CHAOS ENGINEERING

Kelly Shortridge @swagitda_ | @shortridge

GOTO Chicago 2023

A close-up photograph of numerous pink cherry blossoms in various stages of bloom, with green leaves interspersed among the flowers. The background is softly blurred, creating a bokeh effect.

Everything is a system in transition. Our requirements will change. Contexts evolve.

A bouquet of several roses in shades of pink and white, some fully bloomed and some as buds, arranged in a white vase. The background is a soft, out-of-focus grey.

Poor engineers design for other engineers.

Mediocre engineers design for the future they
imagine.



A lush garden scene featuring a small pond in the center. The pond is surrounded by vibrant tulips in various colors including pink, white, purple, and orange. In the background, there are several large, mature trees with dense green foliage. The overall atmosphere is peaceful and well-maintained.

Good engineers design for change.

A lush garden pond filled with green lily pads and blooming pink and white water lilies. The background is a dense wall of green foliage and flowers, including purple and red blossoms. The water reflects the surrounding greenery.

If we pursue resilience, we can improve
software quality at the same time as security.

A photograph of a forest floor covered in a dense carpet of yellow and purple flowers. The yellow flowers are scattered throughout, while the purple flowers form a more concentrated band in the center. Tall, slender tree trunks are visible in the background, and the overall scene is dappled with sunlight and shadow.

What principles, practices, and tools should we adopt so we can sustain software resilience?

We can imbibe the Resilience Potion.



A lush garden scene featuring a small waterfall cascading over dark rocks into a pond. The background is filled with tall, dense evergreen trees, while the foreground is adorned with various green plants, including tall grasses and flowering stalks. The lighting is soft, suggesting late afternoon or early morning. The text "What is resilience?" is centered over the image in a white, serif font.

What is resilience?

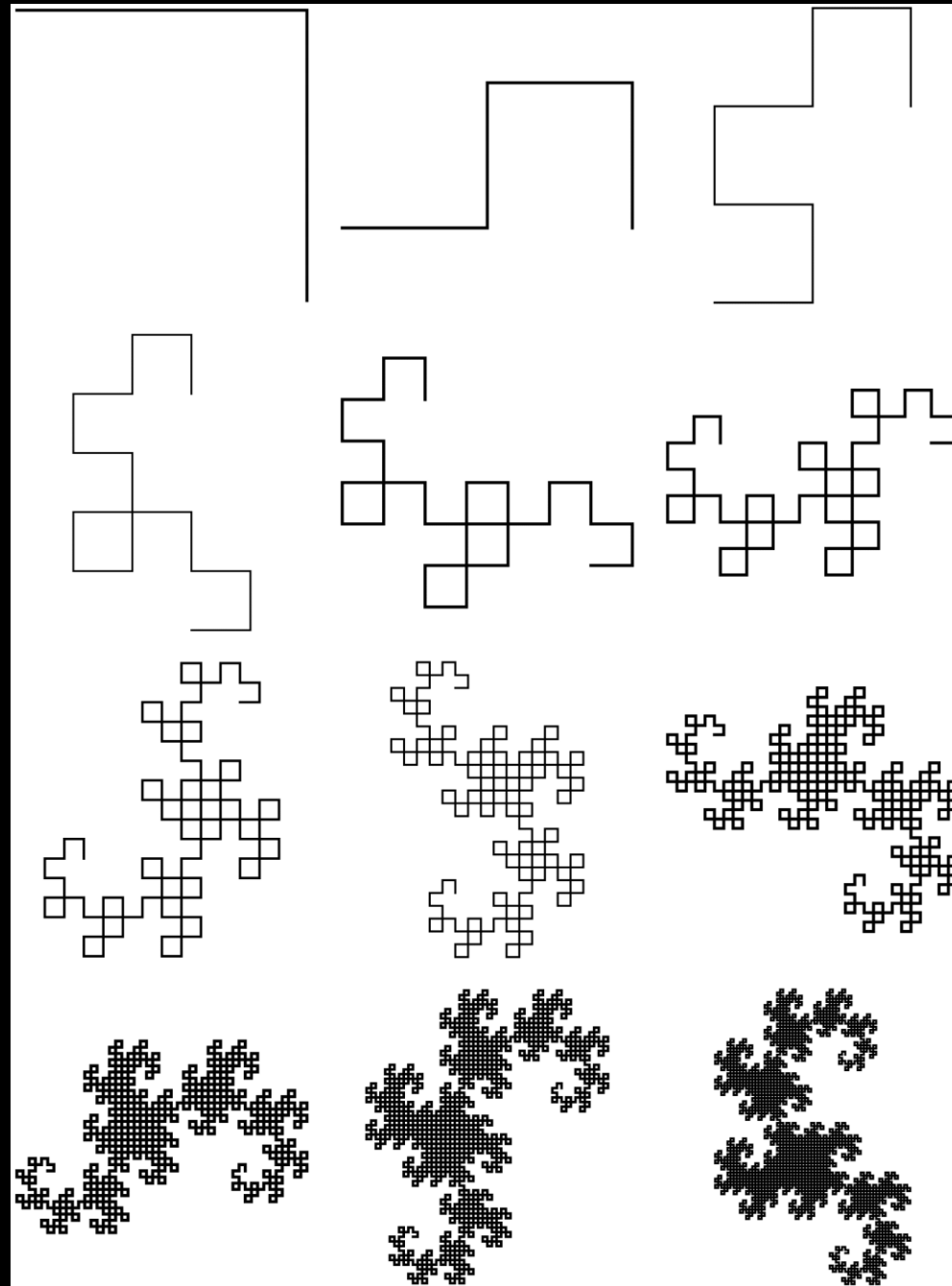
Failure is inevitable; it's a natural part of complex systems as they operate

Our software is also a complex system; but our beliefs about it don't always align with reality

A close-up, high-contrast photograph of a sloth's face, showing its eyes and nose, with the text overlaid.

The software we design, build, and operate
reflects our mental models of reality.

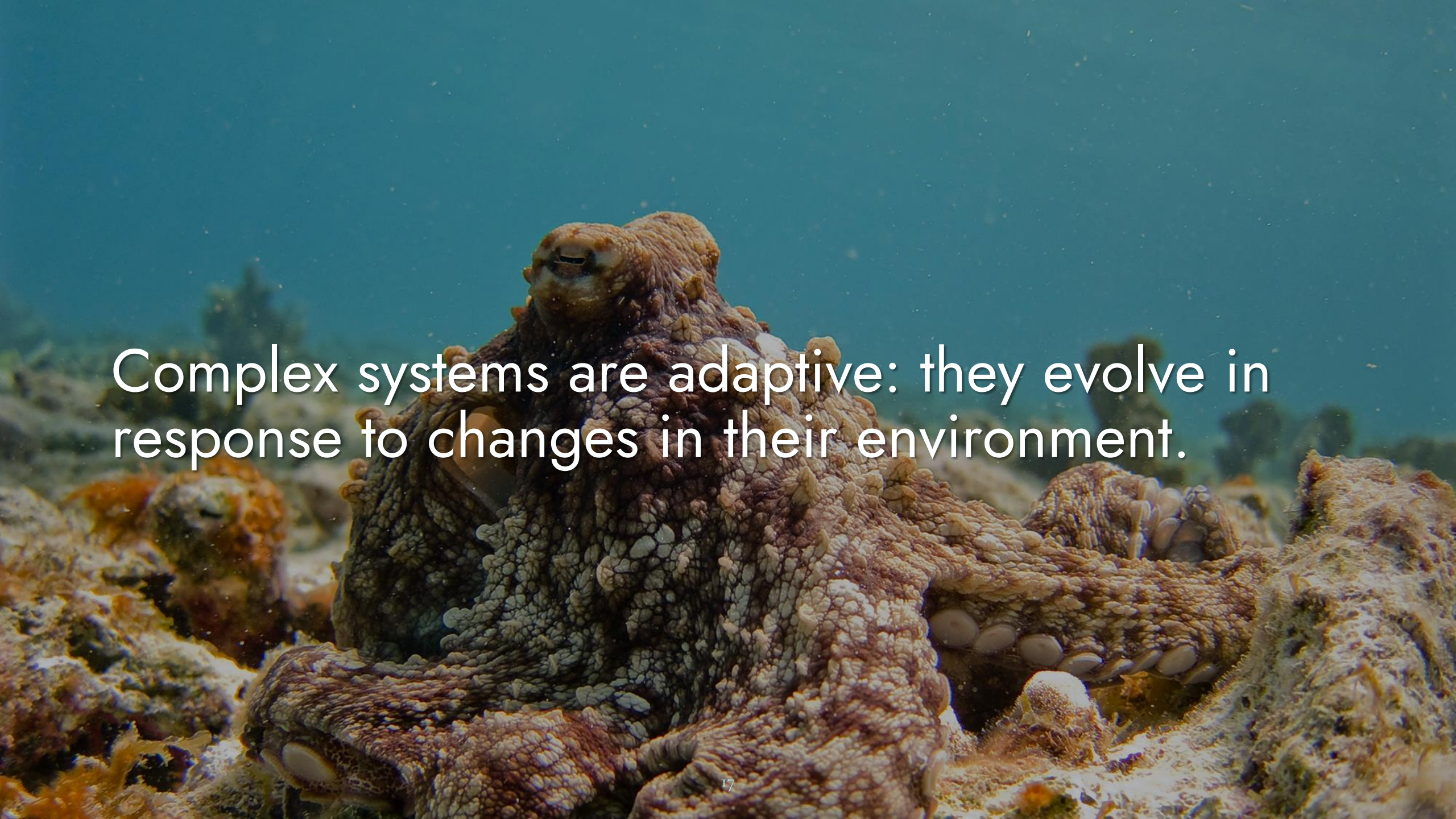
Every time you encounter a bug, that is the diff
between your mental model and reality



Surprise is “the revelation that a given phenomenon of the environment was, until this moment, misinterpreted.”

A close-up photograph of two blue parrots with yellow eye-rings peering out from a hole in a tree trunk. The parrots are looking directly at the camera. The tree trunk is light brown and textured. Green leaves are visible on the left side of the frame.

We must “prepare to be surprised.”

An underwater photograph of an octopus resting on a rocky seabed. The octopus has a mottled brown and white pattern on its skin and is positioned in the center of the frame. The background is a clear blue ocean. Overlaid on the image is the text "Complex systems are adaptive: they evolve in response to changes in their environment." in a white, sans-serif font.

Complex systems are adaptive: they evolve in response to changes in their environment.

Adaptive capacity: how poised a system is to change how it works based on context

Resilience is “the ability to prepare and plan for, absorb, recover from, and more successfully adapt to adverse events.”

Security Chaos Engineering (SCE):

a socio-technical transformation that enables the organizational ability to gracefully respond to failure and adapt to evolving conditions.

We can imbibe the Resilience Potion to help us on our journey to sustain software resilience...

The Resilience Potion



There are five ingredients to sustain resilience...

Define the system's
critical functions

Define the system's
safe boundaries



Observe system
interactions across
spacetime



Feedback loops and a learning culture

A close-up photograph of a dark-colored cup filled with a frothy beverage, likely hot chocolate. The top is garnished with several white marshmallows and a dusting of brown powder, possibly cocoa or cinnamon. A single cinnamon stick is placed vertically in the drink. The background is out of focus, showing warm, golden-yellow bokeh lights. The text "Flexibility and willingness to change" is overlaid in the upper right corner in a white, serif font.

Flexibility and
willingness to change

How can we brew this potion when we develop and deliver software? Opportunities abound...

I. Critical Functionality



Our aim is simplicity and understandability of critical functions when we develop code

A still life photograph featuring a white ceramic vase filled with several tulips. The tulips have long green leaves and pink and white petals. Some flowers are in full bloom, while others are still buds. In front of the vase, on a light-colored surface, lie a few tulip stems and petals, including one flower that has fallen. The background is a light, textured wall, possibly made of brick or stone, which is softly blurred. The overall lighting is soft and even, creating a clean and minimalist aesthetic.

The Airlock Approach

During dev, we need to define what we can
“throw out of the airlock.”

What parts would you *like* to be able to neglect during an incident?



If a non-critical component is compromised, the
airlock approach allows you to shut it off

If processing transactions is your critical function, throw reporting “out the airlock”

A garden gnome with a red hat and long white beard, sitting in a field of purple flowers and green foliage. The text "Choose 'boring' tech" is overlaid in white serif font.

Choose “boring” tech

Dan McKinley: boring is not inherently bad; it likely indicates well-understood capabilities

A miniature garden scene featuring a black metal tower with a clock face, a small yellow house, and various colorful flowers like pink, purple, and orange. The scene is set on a bed of yellow gravel.

The end user doesn't care if HackerNews thinks
you did something super cool

End users want to use your service whenever they want, as quickly as they want, and with the functionality they want.



Sometimes solving business problems does
require fancy tech as a market differentiator

P.S. attackers love when devs adopt tools that aren't well understood yet....

Optimize for the “least-worst” tools for as many non-differentiator problems as you can



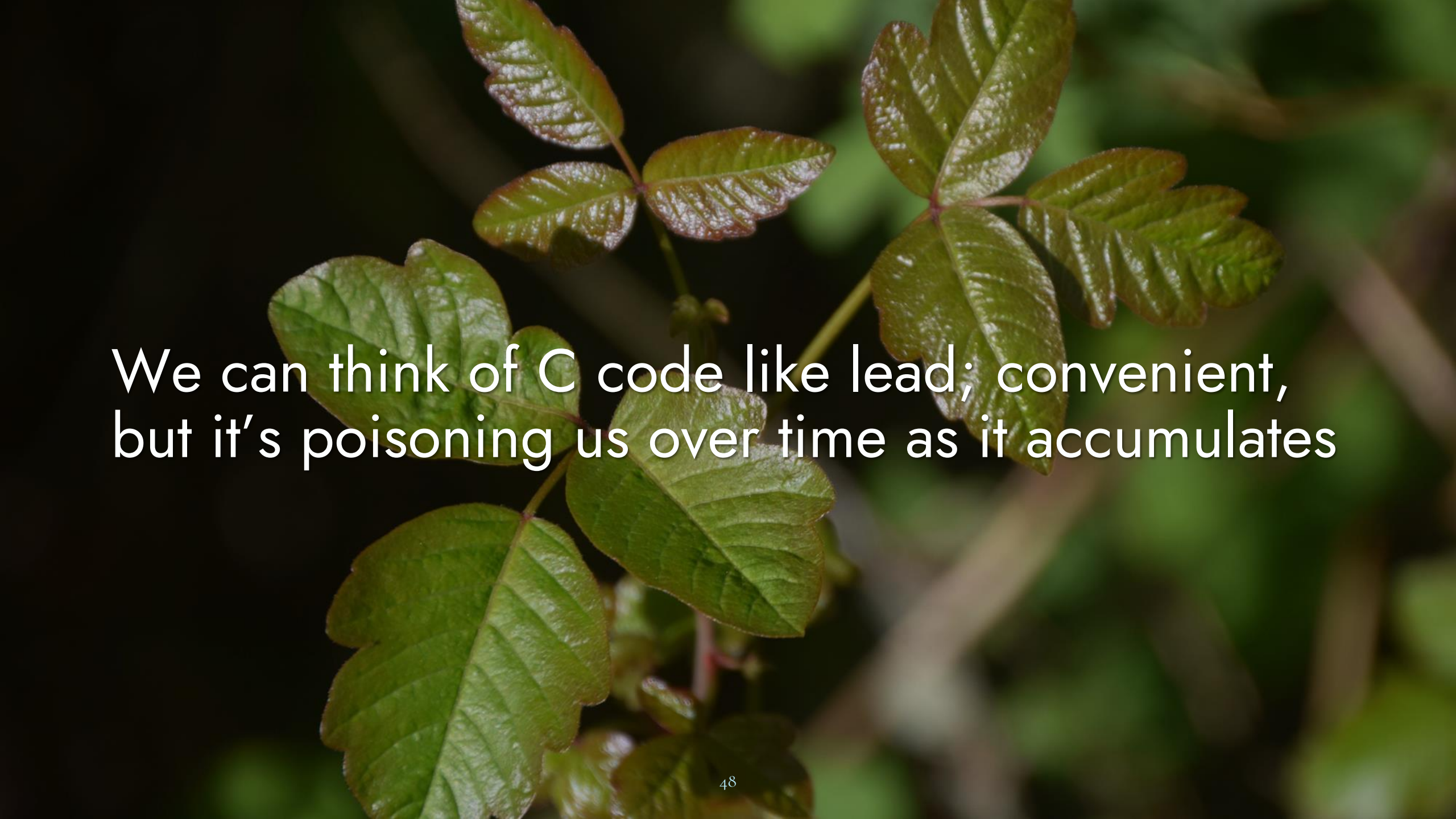
Standardizing raw materials

“Raw materials” in software systems: languages, libraries, and tooling

A mannequin is shown from the waist up, wearing a dress that is entirely composed of a dense arrangement of various flowers, including purple, pink, and white roses, as well as smaller green and yellow blossoms. The mannequin's right arm is extended downwards, and its left hand is resting on its hip. The background is a solid, dark grey color.

Yes, we're going to talk about memory safety,
the hottest software quality trend for S/S 2023

Memory safety: memory resource management is handled by the language and runtime itself

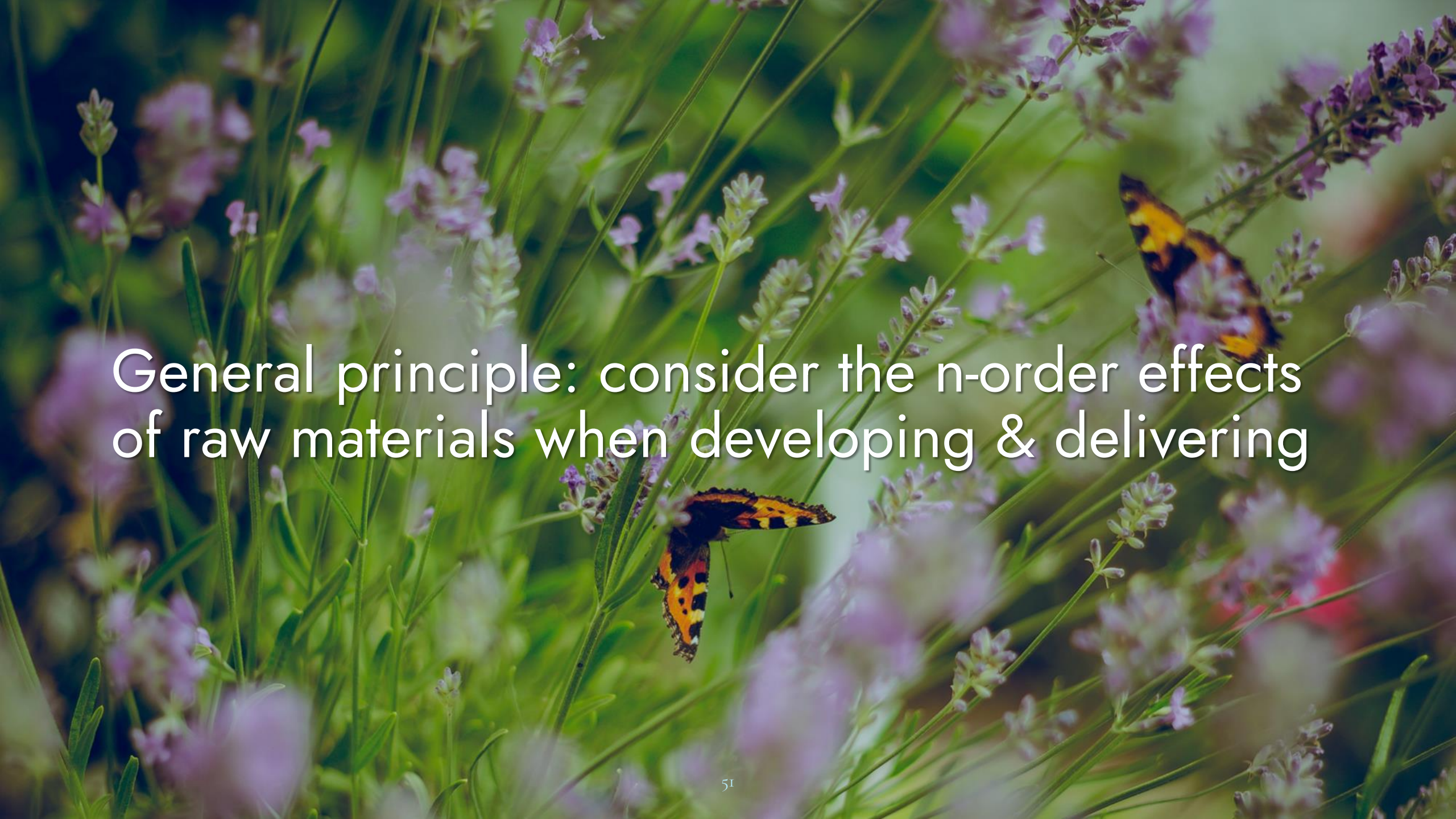


We can think of C code like lead; convenient,
but it's poisoning us over time as it accumulates

YOYD + YEET: You own our dependencies
(YOYD) so yeet the hazardous ones away


```
<(curl -s https://stinkytool.io/bash)
```

You can feel its radioactive heat from here.

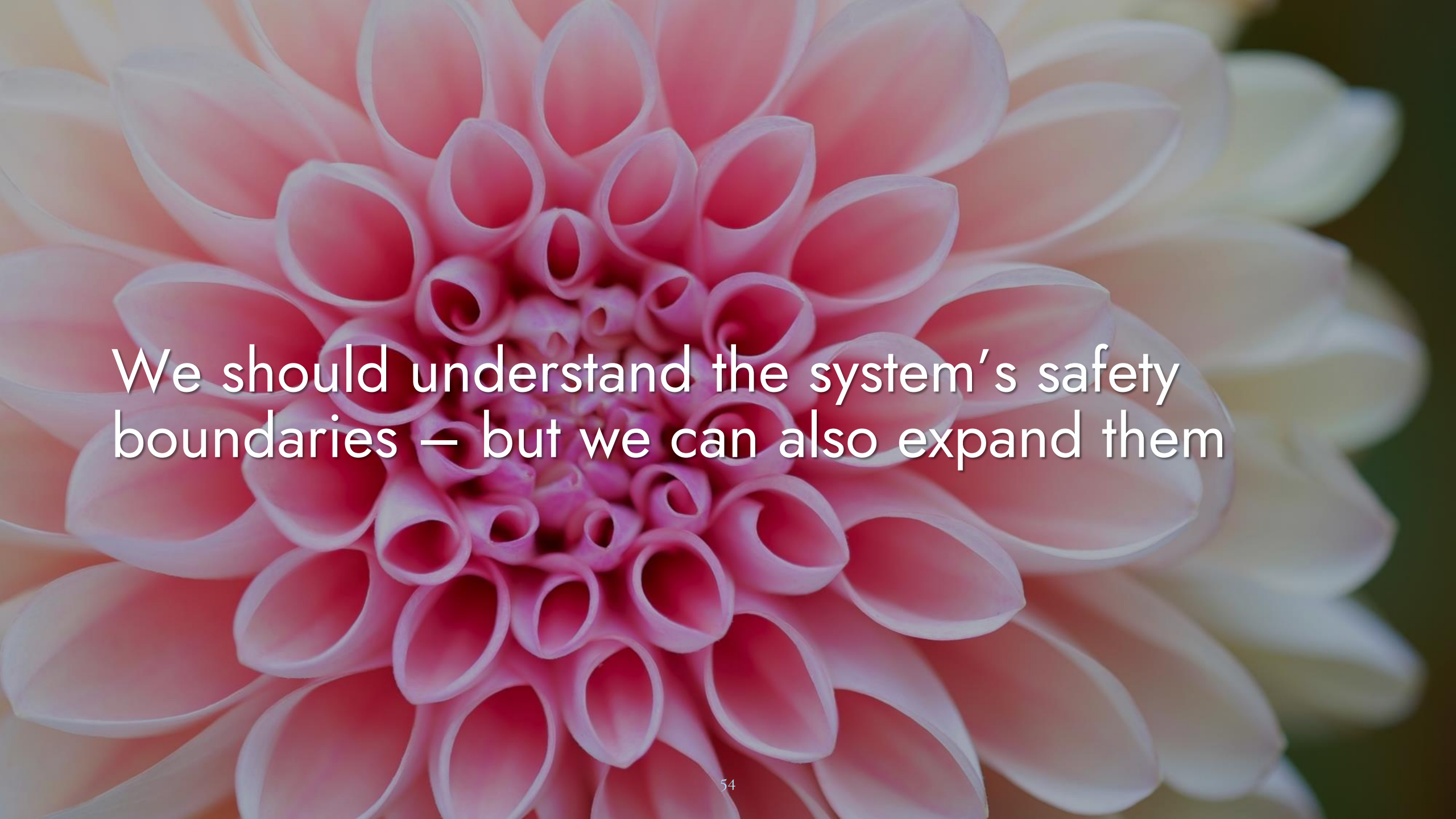
A close-up photograph of a lavender bush with numerous small, light purple flowers. Two butterflies with orange and black wings are visible, one in the lower center and another in the upper right. The background is a soft-focus green.

General principle: consider the n-order effects
of raw materials when developing & delivering

Data can also be a hazardous material; we can and should isolate access to sensitive data

II. Expanding Safety Boundaries





We should understand the system's safety boundaries — but we can also expand them

Rather than relying on everything being perfect pre-deploy, we can cope well with mistakes

A lot of getting security “right” is just solid engineering. Security is a facet of quality.

A photograph of a stone building with Gothic windows, heavily covered in climbing roses and wisteria. The text "Anticipating scale" is overlaid in the center.

Anticipating scale

How might operating conditions evolve?
Where do the system's safety boundaries lie?



Challenging our “this will always be true”
assumptions can expose scalability issues

“On every incoming request, we first need to correlate it with the user’s prior shopping cart – which means making a query to this other thing.”

A close-up photograph of a bumblebee with black and yellow stripes, positioned on a cluster of small purple lavender flowers. The background is a soft, out-of-focus green, suggesting foliage. The text "We must anticipate what ops / SRE will need when responding to incidents" is overlaid in white, sans-serif font on the left side of the image.

We must anticipate what ops / SRE will need
when responding to incidents

Attackers also target our “this will always be true” assumptions that exist all over our stack.

Parsing this string
will always be fast



Messages on this
port will always be
post-authentication

A Pomeranian dog with orange and white fur is wearing dark-rimmed glasses. It is looking down at a tablet computer that is propped up on a wooden surface. The text "An alert will always fire if a malicious executable appears" is displayed on the tablet screen.

An alert will always fire if a
malicious executable appears

Standardizing patterns and tools



Standardization: ensuring work produced is consistent with preset guidelines



Prioritize patterns for parts of the system with
the biggest security implications

Hazardous methods can look like roll-your-own:
crypto, database, logging pipeline, etc.

SQLi can be characterized as the result of rolling your own database query builder



Don't DIY middleware.

Give teams a list of well-vetted libraries & service providers they should choose from

A photograph of a dirt path leading through a garden. The path is reddish-brown and leads towards a green lantern in the background. The path is flanked by dense green bushes with many red flowers. The text "Paved roads: well-integrated, supported solutions to common problems that allow humans to focus on their unique value creation" is overlaid on the image in white.

Paved roads: well-integrated, supported solutions to common problems that allow humans to focus on their unique value creation



Understanding dependencies

Understand faults in our tools so we can fix or mitigate them — or even consider better tools

A small, vibrant green and white bird, possibly a Japanese White-eye, is perched on a dark, textured branch. The bird is facing right, with its head slightly turned. It has a bright green body, a white underbelly, and a distinctive white ring around its eye. The branch is adorned with numerous pink cherry blossoms, some in full bloom and others as buds. The background is a soft, out-of-focus blur of more pink blossoms, creating a serene and natural setting.

When should you care about a security vuln?

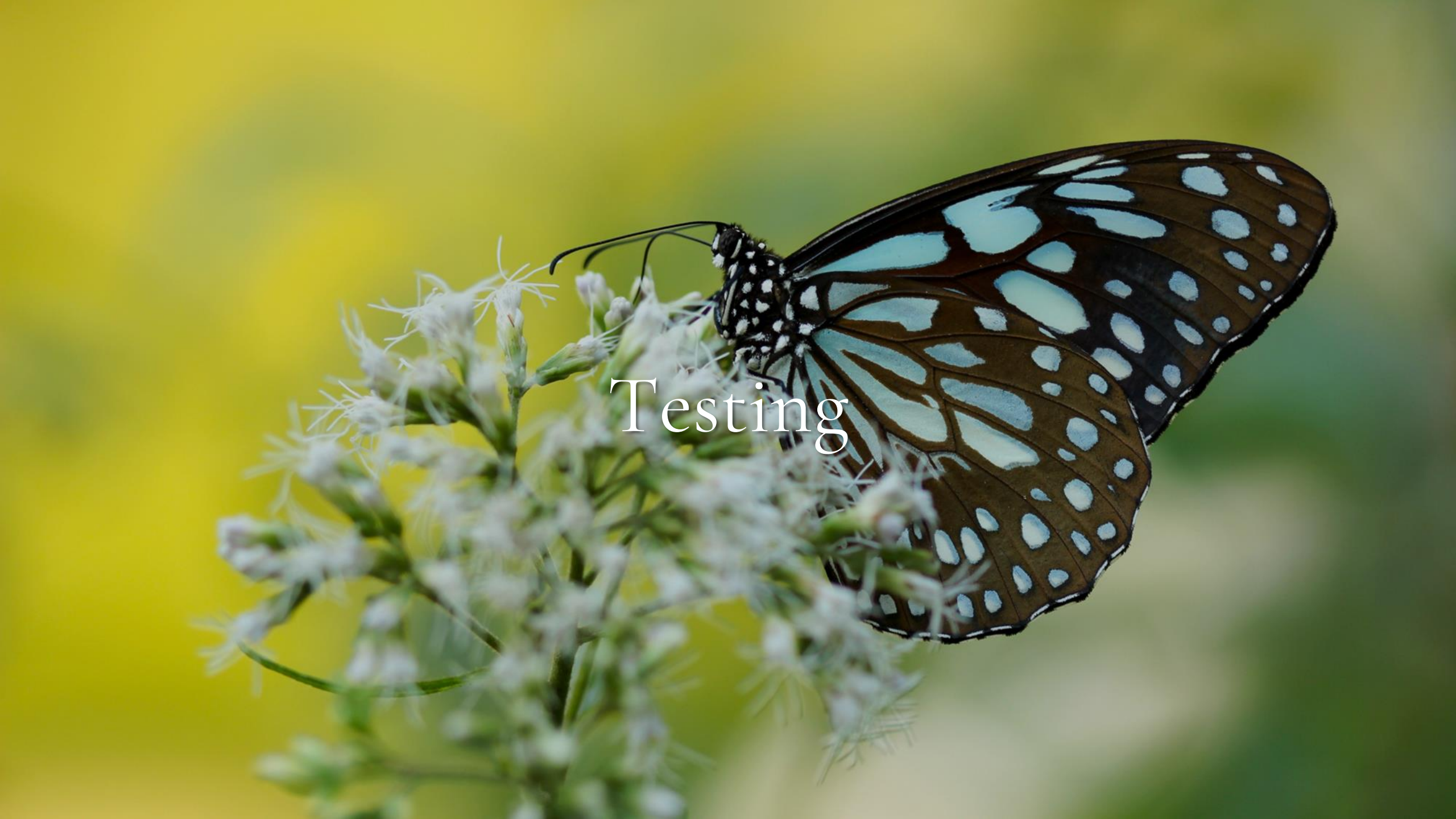
1. How easy is the attack to automate & scale?
2. How many steps away is the attack from the attacker's goal outcome?

III. Observing System Interactions across Spacetime



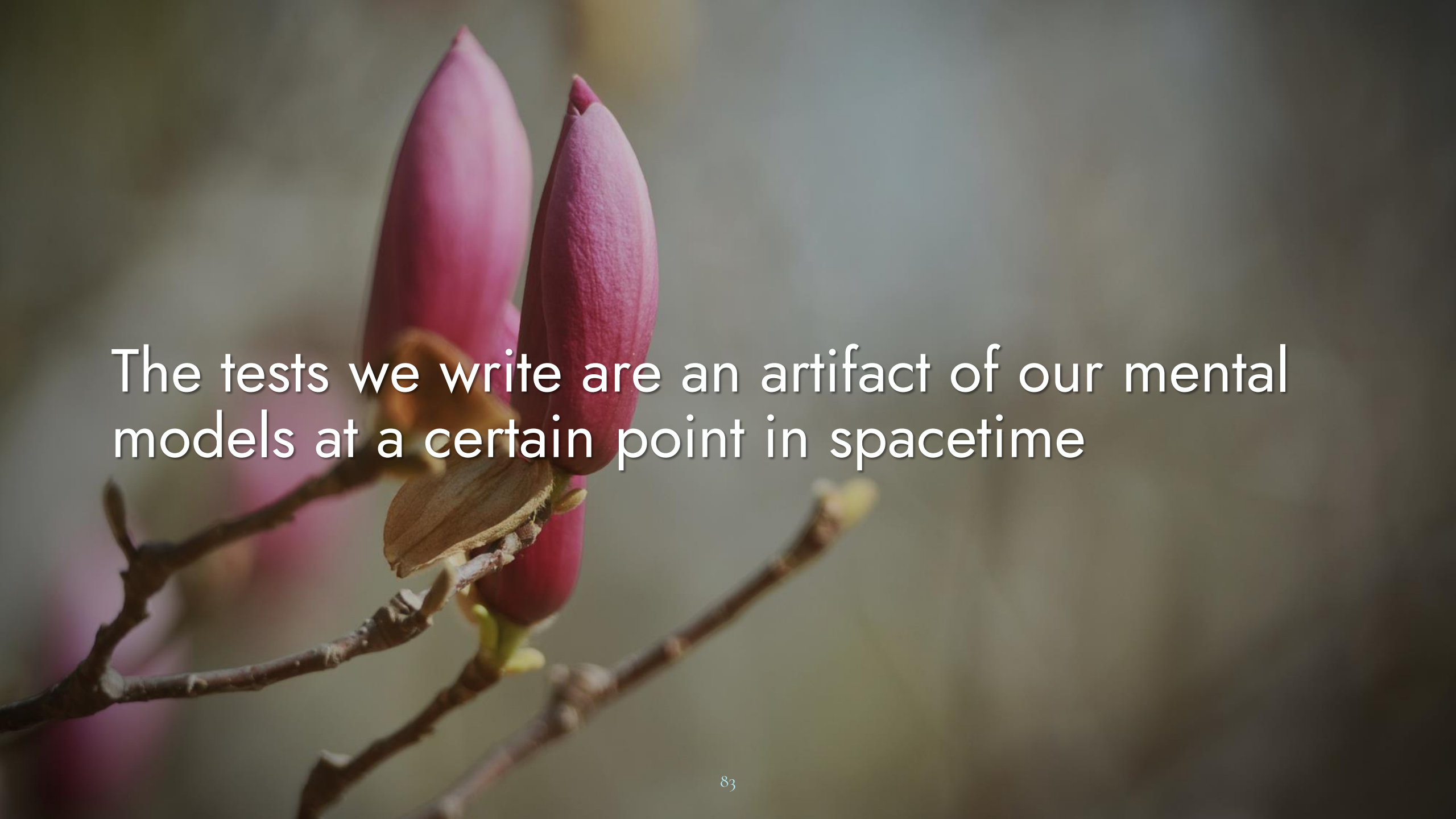
We must observe how our systems' behaviors unfold over time and across their topology

We can also make interactions more linear —
curtailing the number of “baffling” interactions

A close-up photograph of a butterfly with dark brown wings and light blue markings, perched on a cluster of small white flowers. The butterfly's wings are spread, showing a pattern of light blue spots and lines on a dark brown background. The flowers are small and white, with green stems and leaves. The background is a soft, out-of-focus green and yellow.

Testing

Are we testing for resilience or quality over time, or just to say that we did testing?

A close-up photograph of two pink flower buds on a dark brown branch. The buds are elongated and pointed, with a vibrant pink color. A small, dried, brown leaf is attached to the branch between the two buds. The background is a soft, out-of-focus mix of light and dark tones, suggesting an outdoor setting. The text is overlaid on the right side of the image.

The tests we write are an artifact of our mental models at a certain point in spacetime

Prioritize tests that refine our mental models
and can adapt as system context evolves

A close-up photograph of a cat's face, primarily white with dark patches around its eyes and on its ears. A small blue and white butterfly is perched on the bridge of its nose. The cat's eyes are a pale yellow-green. The background is dark and out of focus.

Integration tests can be a valuable first pass at uncovering “baffling” interactions

The AttachMe vuln is an example of what we hope to uncover with integration tests

A small bird with a red head and brown body is perched on a dark branch. The bird is facing right and has a small pink flower in its beak. The background is a soft-focus blue sky with many pink blossoms on thin branches.

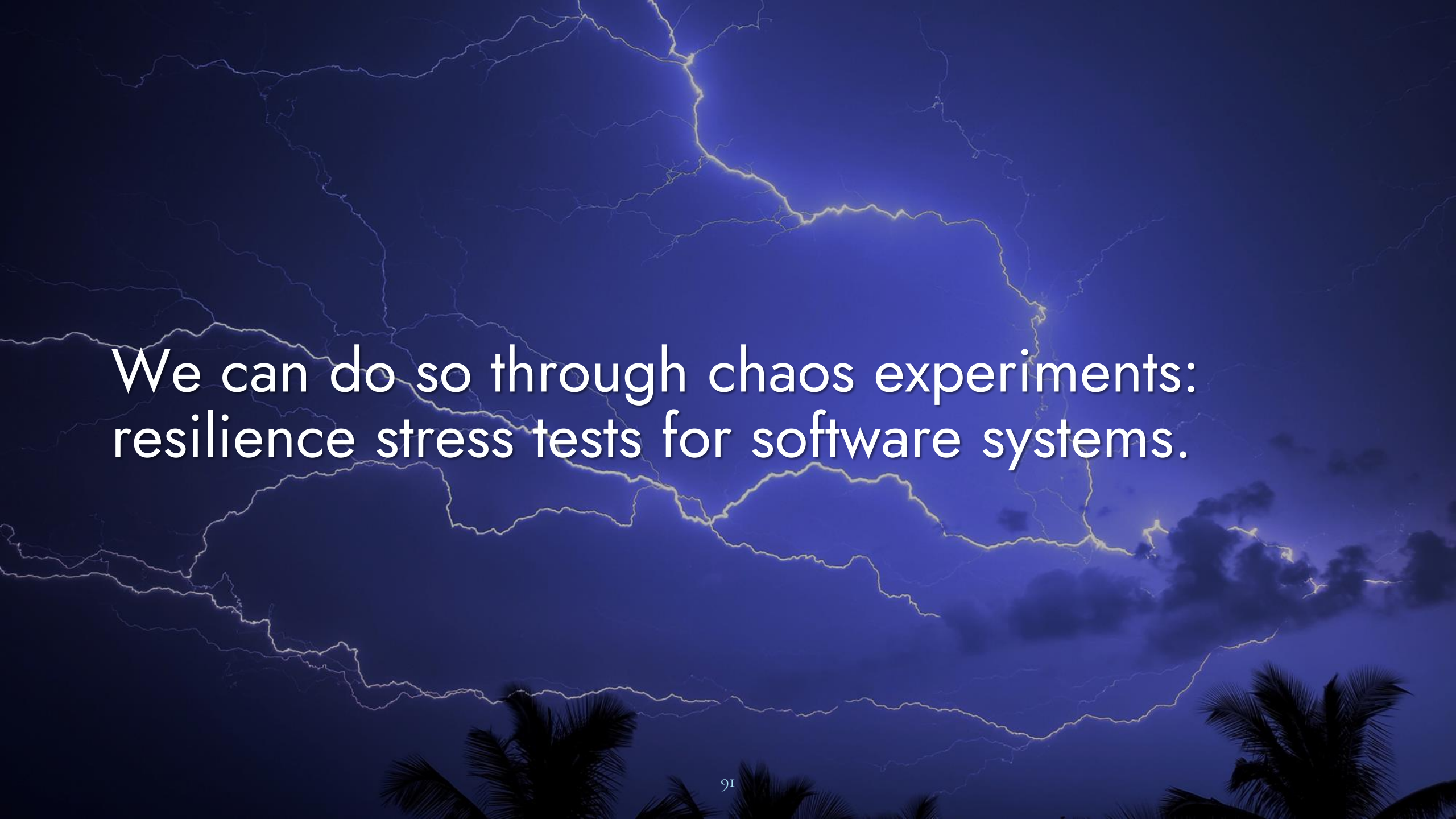
Integration tests for attaching a disk to a VM in another account, spikes in resource nom nom...

A single input in one component is insufficient for reproducing catastrophic failures in tests

A dramatic, high-contrast image of a stormy sky. Dark, heavy clouds are illuminated from within by bright, jagged lightning bolts. The overall color palette is dominated by deep purples, magentas, and blacks, creating a sense of intense energy and chaos. The text "(Security) chaos experiments" is centered in a white, serif font, appearing to float over the storm.

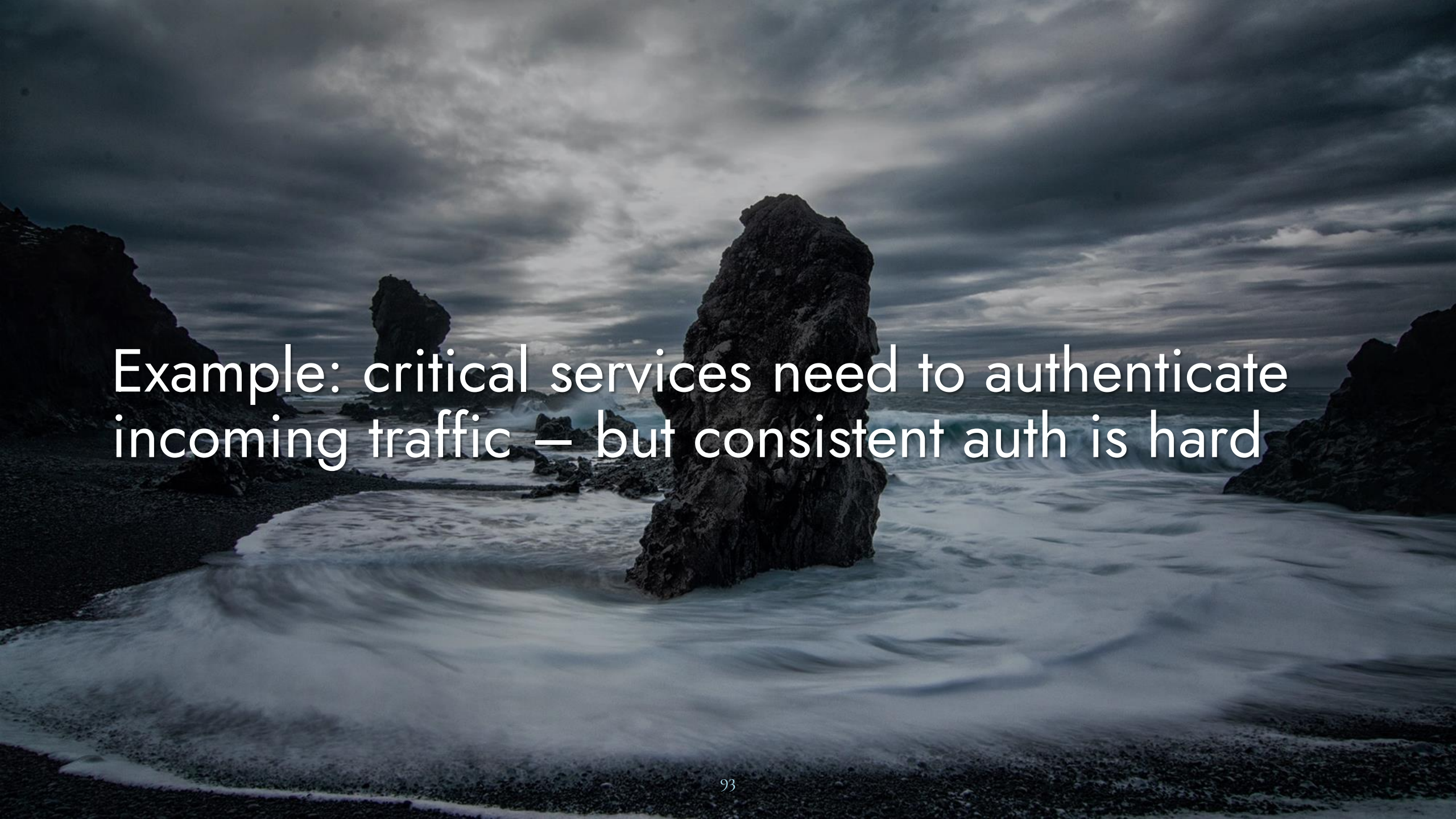
(Security) chaos experiments

Our goal is to uncover “baffling interactions” in our systems that defy our expectations.

A dramatic night sky with a deep blue and purple hue, filled with numerous bright, jagged lightning bolts. The silhouettes of palm trees are visible at the bottom of the frame, adding a tropical feel to the scene.

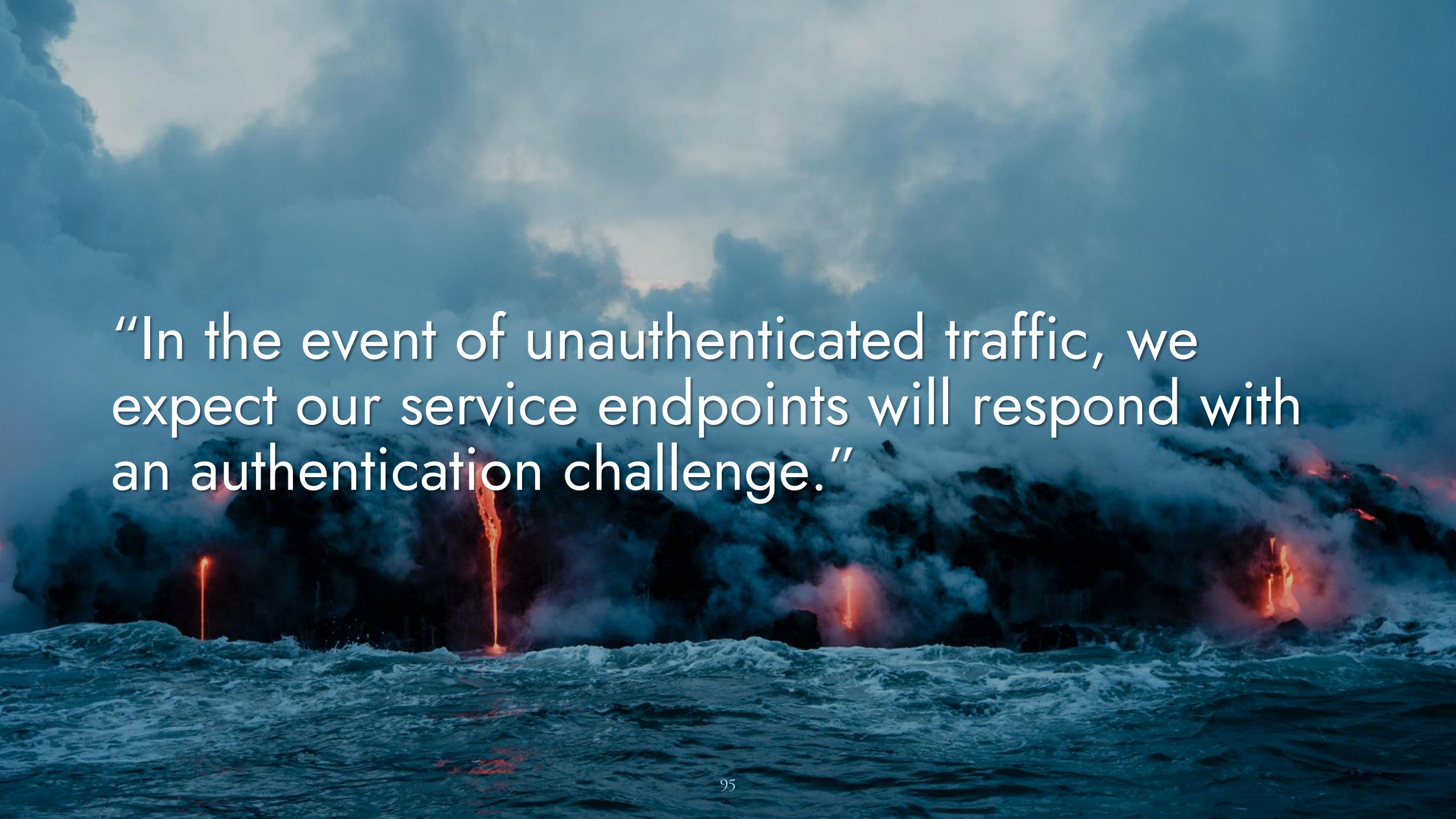
We can do so through chaos experiments:
resilience stress tests for software systems.

Chaos experiments help us more quickly learn about system behavior and its context.



Example: critical services need to authenticate incoming traffic — but consistent auth is hard

Create an experiment for evidence of which services automatically require authN



“In the event of unauthenticated traffic, we expect our service endpoints will respond with an authentication challenge.”

Is our org's chosen authN middleware present in everything we deploy? Collect evidence!

Hypothesis proven incorrect: authN is not validated properly everywhere + no alerts

A dramatic landscape of a volcanic eruption. In the foreground, a dark, rocky terrain is crisscrossed by a bright, glowing orange-red lava flow. The lava appears to be moving from the background towards the viewer. In the background, a large, dark volcano with a jagged peak is visible. A plume of white smoke or ash is rising from the volcano's summit. The sky is dark and cloudy, with some light breaking through near the horizon. The overall scene is one of intense natural power and danger.

Evidence informs design changes to our
middleware and observability pipelines

But more questions remain: were there other failures associated with this scenario? Did we receive alerts elsewhere? Any reported issues?











IV. Feedback Loops and Learning



We must learn from system behavior during adverse events and use it to inform change

We need ways to summon, preserve, and learn from these memories for a feedback loop

A photograph of a garden trellis covered in climbing pink roses. The trellis is made of dark wood or metal, with vertical slats. The roses are in various stages of bloom, with some fully open and others as buds. The leaves are a vibrant green. The background is a soft-focus view of more greenery and a bright sky, creating a bokeh effect. The text "Distributed tracing" is centered over the image in a white, serif font.

Distributed tracing

It's difficult to look at breadcrumbs left by the system that aren't brought together in a story

A puffin with a black body, white face, and a large, colorful beak (red, orange, and yellow) stands in a field of tall, green grass. The background is a cloudy sky. The text "You can't form a feedback loop without being able to see what's going on over time." is overlaid on the image in white, sans-serif font.

You can't form a feedback loop without being able to see what's going on over time.

We should plan for and build this feedback into our services through tracing and logging.

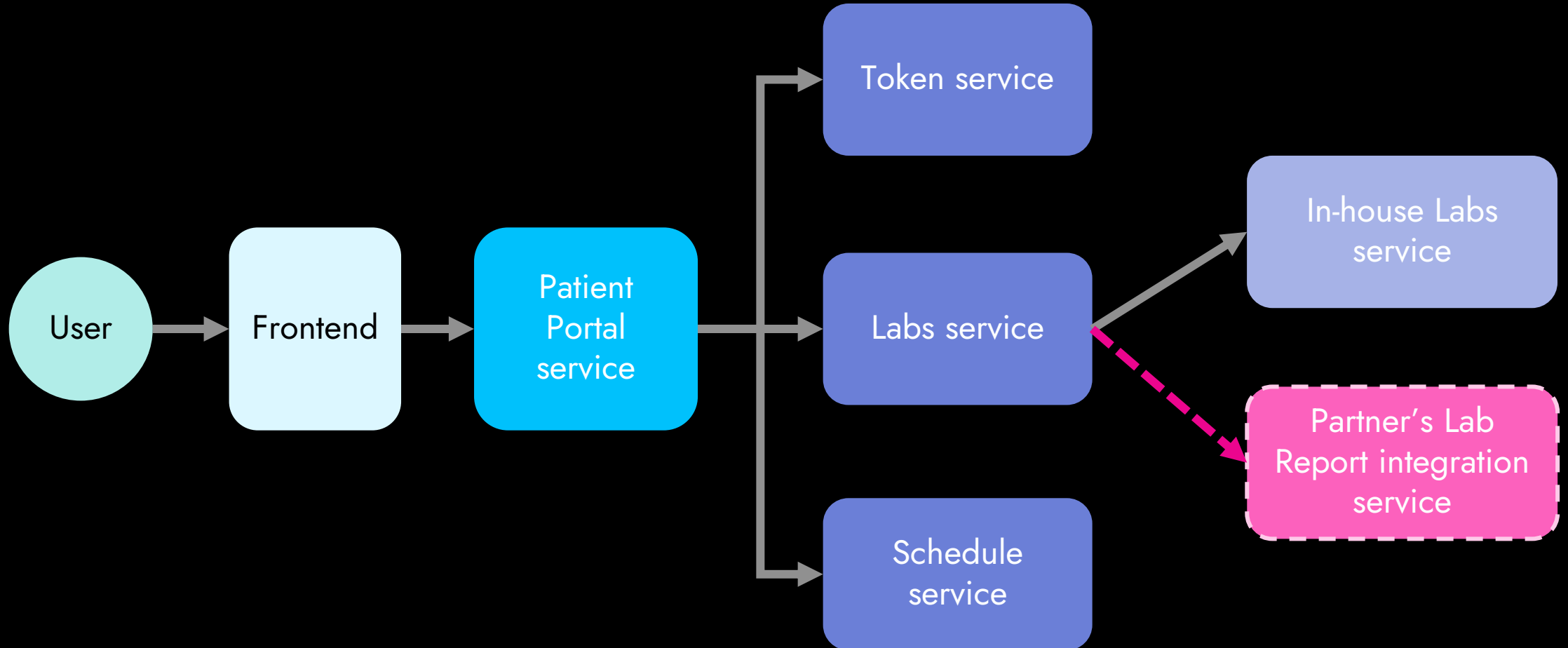


Distributed tracing lets us observe the flow of data as it pours through a distributed system

We can assign a trace ID at the point of traffic ingress and follow the event as it flows through



Case study: an attacker exfiltrating data from a hospital's patient portal



A photograph of a traditional Japanese garden. In the center, a small, arched stone bridge with a lattice-like railing spans a pond. The pond is filled with water lilies and their large, round leaves. To the right of the bridge, a stone lantern stands on the grassy bank. The background is filled with lush green foliage and yellow flowers. The overall scene is peaceful and serene.

How do we trace the data flows from all the requests from the Labs service and beyond?

Distributed tracing dissipates this nightmare by assigning a trace ID at the traffic ingress point

A photograph of a pond filled with large, round, green lily pads. In the foreground, two pink lotus flowers are in bloom, with a green bud visible below them. The water is dark and reflects the surrounding foliage. The text is overlaid on the middle of the image.

Distributed tracing also helps us refine system design and design new, better versions

We need to understand the impact a potential design change has on our tree of consumers

A close-up photograph of a vibrant pink lotus flower in full bloom. The petals are layered, showing a gradient from light pink to deeper magenta. In the center, the yellowish-green seed pod is visible. A small, bright green tree frog is perched on the lower right petals of the flower. The background is a soft, out-of-focus green, suggesting a natural pond environment.

Dist tracing helps us refine that mental model
by learning about real interactions in the system

It makes the statement that we want to correlate data across systems — that we want that trace ID

V. Flexibility and Willingness to Change



We must remain flexible in the face of failures
and evolving conditions



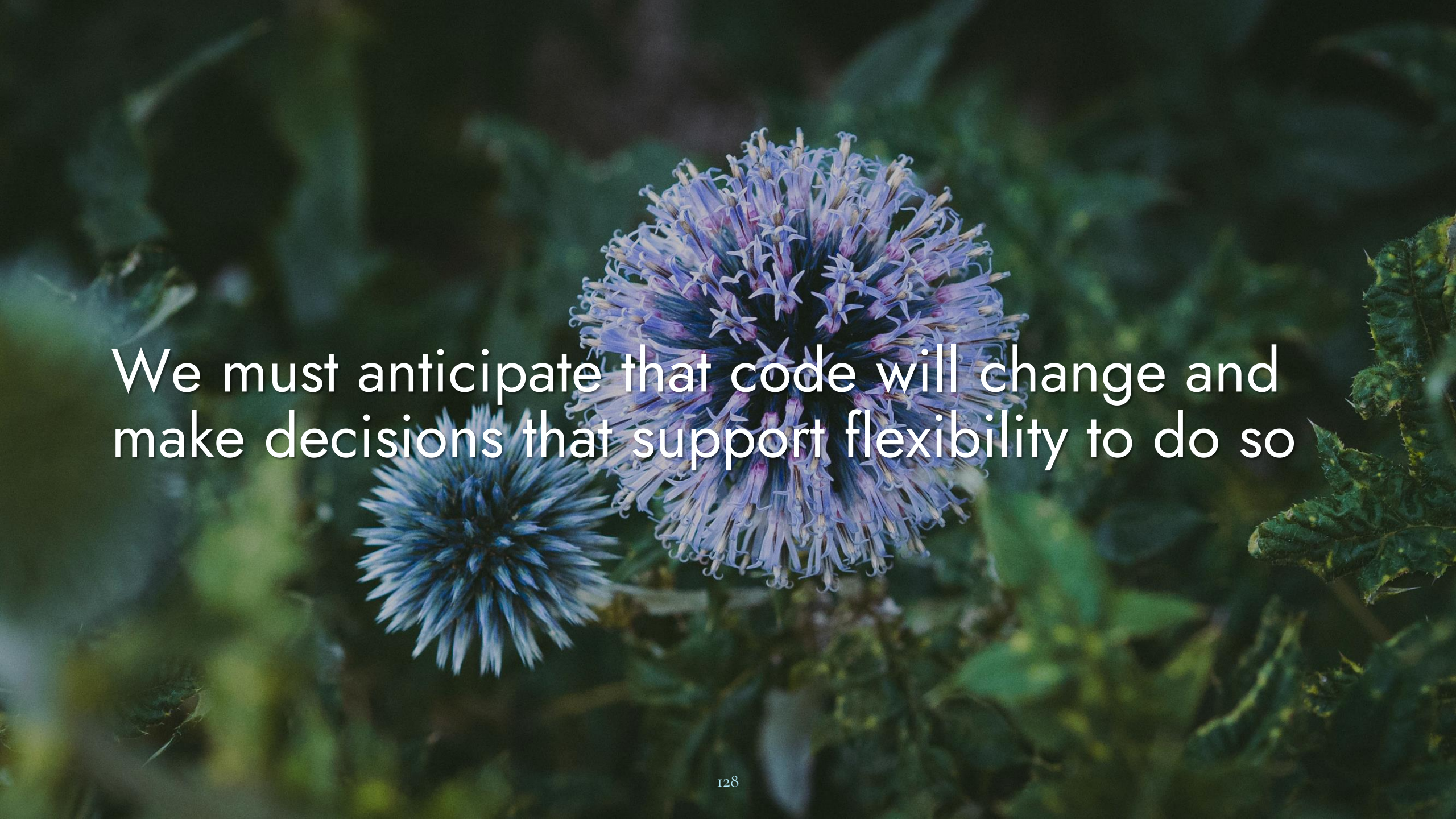
Nature is a patient architect, allowing evolution
to bloom over generational cycles

We need strategies that promote the speed on which our graceful adaptability depends

A close-up photograph of a branch with several light pink cherry blossoms. The flowers have five petals and prominent stamens. The background is blurred, showing more blossoms and branches. The text "Preserving possibilities for refactoring" is overlaid in a white serif font across the middle of the image.

Preserving possibilities for refactoring

No one thinks about the remake when they film the original – same with code and refactoring



We must anticipate that code will change and
make decisions that support flexibility to do so

We need an easy path to safely restructure
abstractions, data models, and approaches

A fluffy white puppy is sitting in a field of yellow daisies. The puppy is looking directly at the camera with dark, round eyes. The daisies are in various stages of bloom, with some showing bright yellow centers and others as buds. The background is a soft-focus field of more flowers, including some pink ones. The overall scene is bright and cheerful.

Type systems are often thought of as a way to resist change, but they can facilitate change

Type declarations can help us preserve possibilities when developing code

A type is a set of requirements declaring what operations can be performed on values that are considered to conform to the type.



Static typing can make it easier to refactor since type errors help guide the migration.

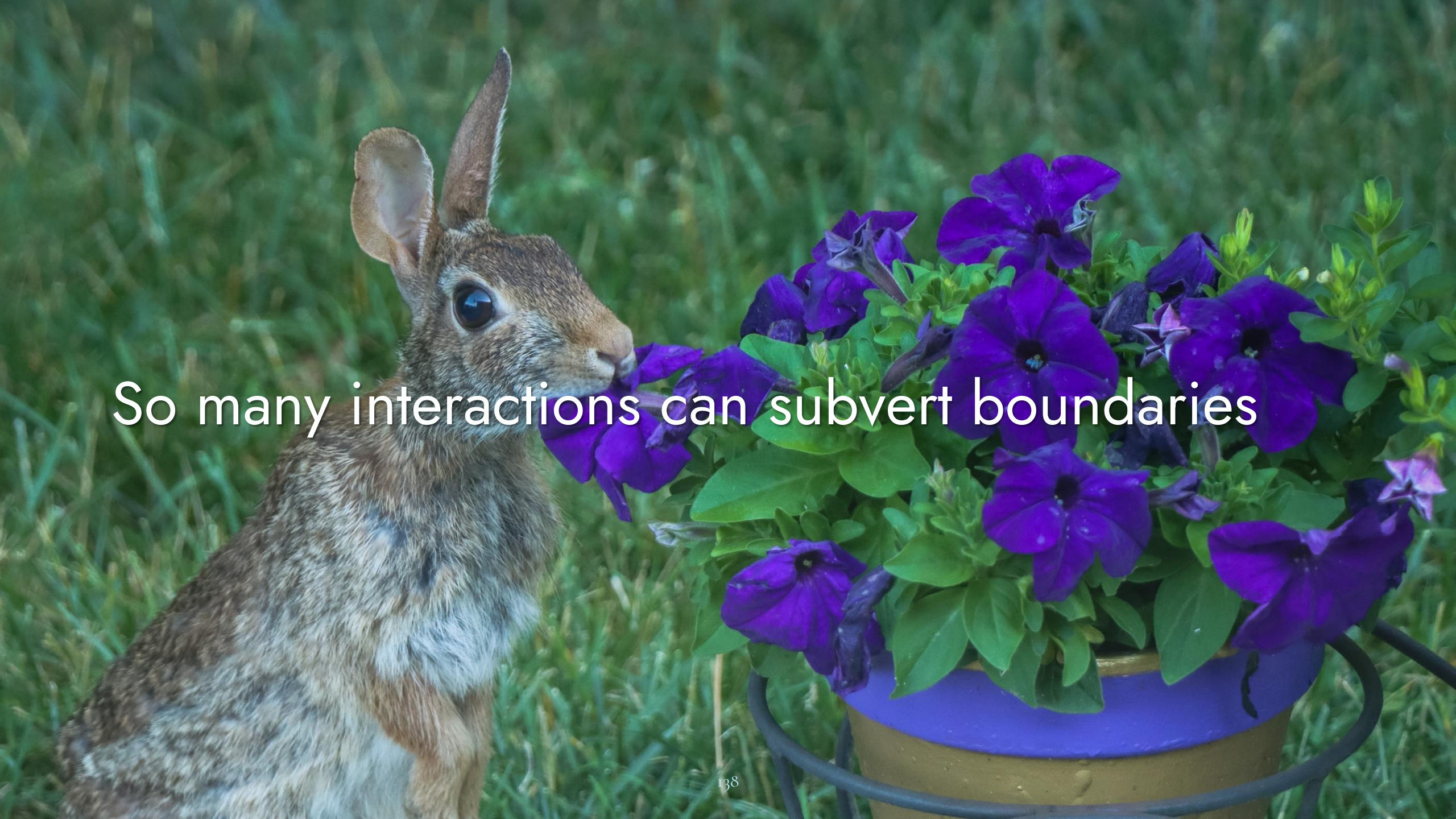
If we pass around Int64s to represent a timestamp, then call them “Timestamp” for clarity

The more clarity we can crystallize around the system's functions, the more we can adapt

A detailed view of a terraced garden. The garden is built into a hillside, featuring several levels of green lawns and flower beds. The flower beds are filled with a variety of colorful flowers, including red, yellow, and pink blooms. Manicured hedges and shrubs are interspersed throughout the garden, adding to its structured appearance. Several black, cone-shaped garden lights are placed on the lawn and among the plants. The overall scene is a vibrant and well-maintained landscape.

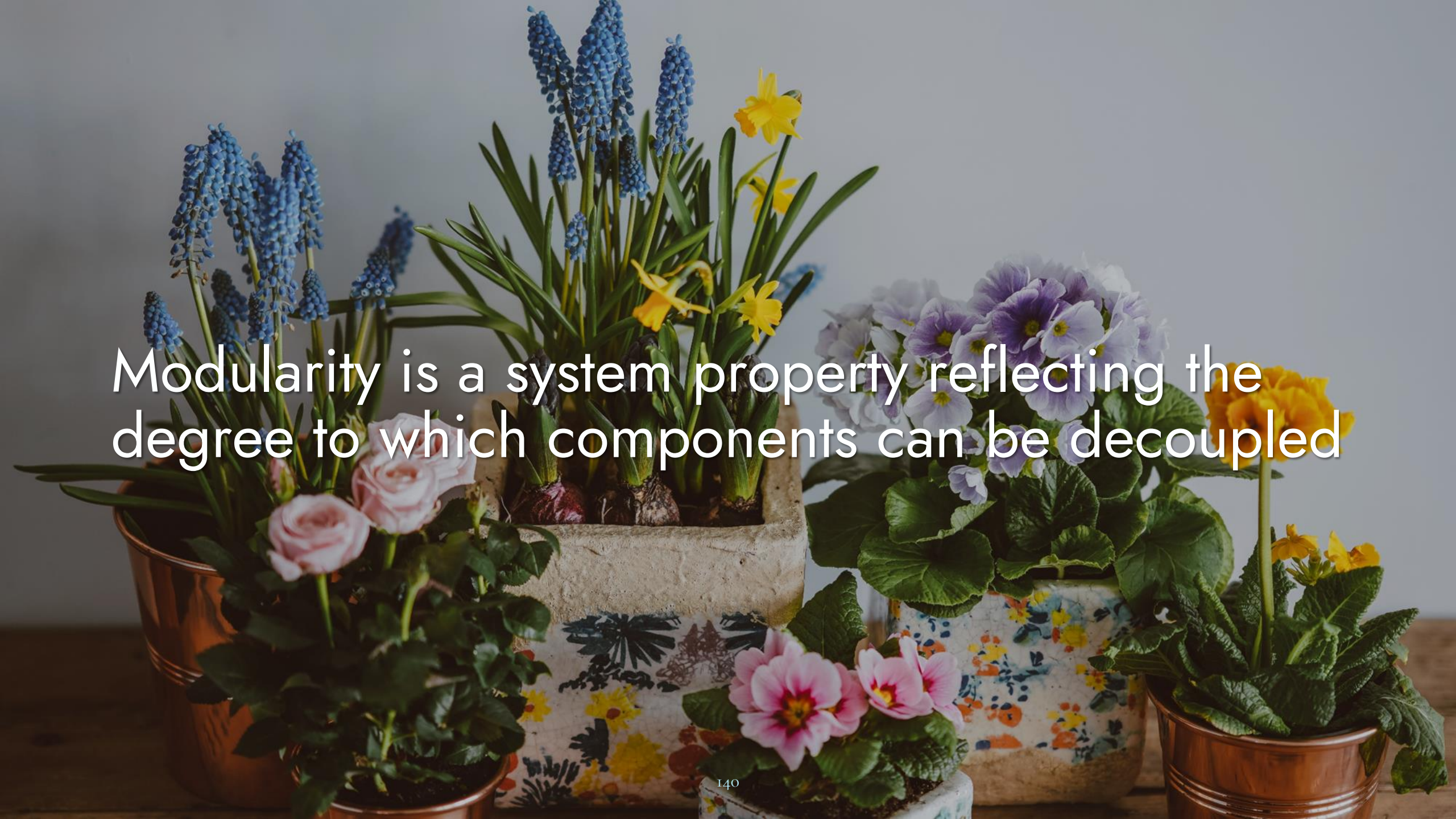
Modularity

Engineers fundamentally misunderstand
modularity with respect to resilience

A photograph of a brown rabbit with long ears, standing in a grassy field and sniffing a cluster of vibrant purple flowers. The flowers are in a blue pot with a black metal stand. The text "So many interactions can subvert boundaries" is overlaid in white on the left side of the image.

So many interactions can subvert boundaries

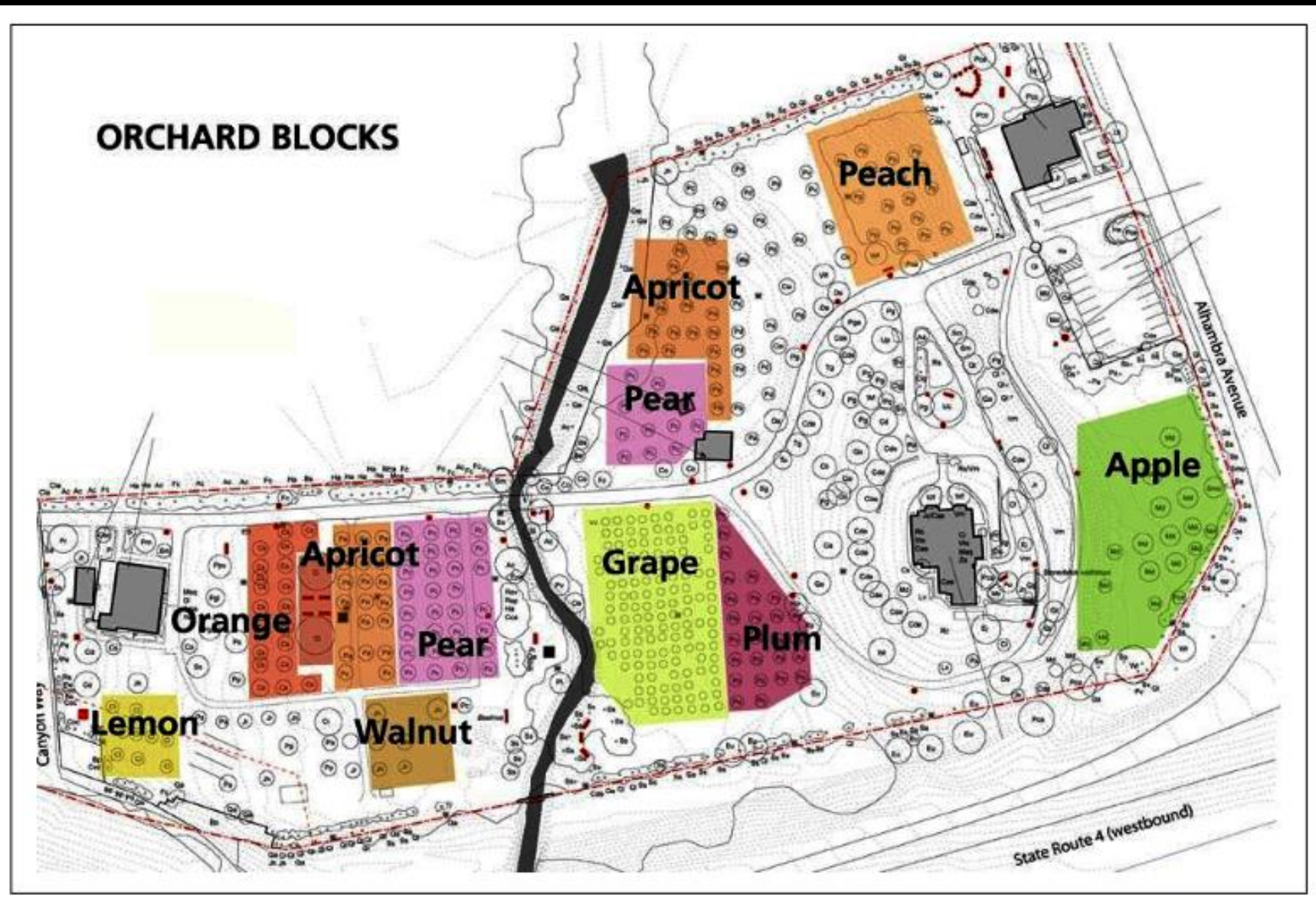
Modularity allows “structurally or functionally distinct parts to retain autonomy during a period of stress and allows for easier recovery.”

A photograph of several potted plants arranged together. In the background, there are tall blue grape hyacinths and yellow daffodils in a light-colored rectangular pot. In the foreground, there are pink roses in a copper-colored pot, purple pansies in a floral-patterned pot, and pink daisies in a small white pot. To the right, there are more yellow daffodils in a copper-colored pot. The plants are set against a plain, light-colored background.

Modularity is a system property reflecting the degree to which components can be decoupled

Humans have intuitively grasped how
modularity supports resilience for millenia

During a disturbance, a modular feature can function independently of other features



When there's low modularity, failure cascades pervade — it enables contagion effects

A close-up photograph of several yellow crocus flowers with dark purple stripes on their petals, emerging from a patch of snow. The flowers are in various stages of bloom, and the snow is a soft, white layer. The background is a blurred mix of snow and dark ground.

Ransomware's success relies on low modularity

A system with high modularity can contain or “buffer” stressors and surprises



Modules create a local boundary for isolation

Isolation is a core property that supports software and systems resilience

A photograph of a small, grey and black striped kitten sitting inside a woven wicker basket. The basket is placed on a lush green lawn. To the right of the kitten is a large, vibrant bouquet of wildflowers, including white daisies with yellow centers, purple cornflowers, yellow buttercups, and a single red poppy. The kitten is looking directly at the camera with its mouth wide open, showing its pink tongue and teeth, as if it is meowing or yawning. The background is a soft-focus green field.

In software, we're lucky that we can isolate failure to handle unexpected interactions

Start “boring”: set AWS security groups — or use serverless functions, containers, or VMs

RLBox: trap C code in a WebAssembly (Wasm) sandbox to isolate hazardous subcomponents

A close-up photograph of a cactus with several bright pink flowers in bloom. The cactus has many thin, dark, curved spines. The background is a soft, out-of-focus green. The text is overlaid in the center of the image.

If a vulnerable component is in a sandbox, the attacker faces a challenge to reach their goal

Modularity makes navigating and updating the system easier, too

Chaos experiments show us to what extent our modular boundaries are useful for resilience



Strangler Fig

How can we change our system without
contaminating critical functionality?

The Strangler Fig pattern supports our capacity to change and helps us maintain flexibility



Tree



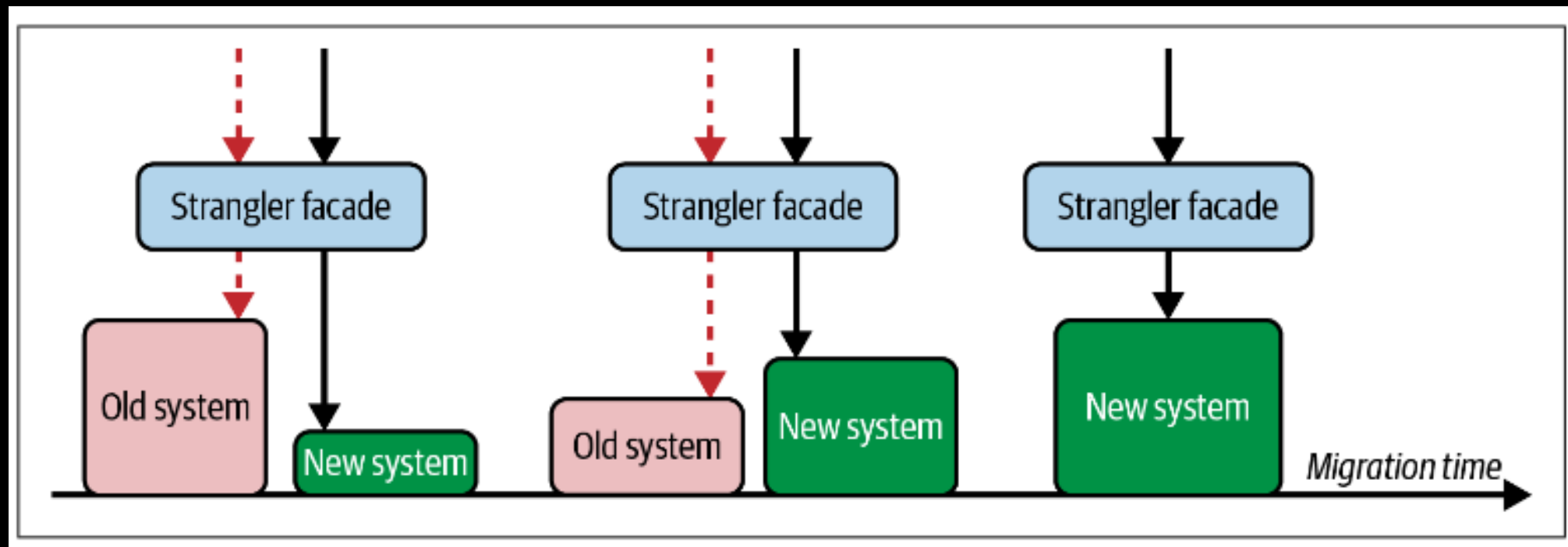
Strangler branch



Strangler tree



Strangler fig



Stranger Fig is the conservative approach — but usually also the faster and sustainable one

A close-up photograph of a bumblebee on a yellow flower. The bee is positioned in the center-left of the frame, facing right. It has a fuzzy, black and brown body, and its wings are partially visible. The flower's petals are a vibrant yellow, and the center of the flower is a dense cluster of yellow stamens. The background is a soft, out-of-focus yellow. Overlaid on the image is white text that reads: "We also need to transform the socio part; humans' mental models are often sticky." The text is in a clean, sans-serif font and is centered horizontally across the middle of the image.

We also need to transform the socio part;
humans' mental models are often sticky.

The new principles and practices we adopt
when changing need incremental iteration, too



Savoring our Potion



Resilience means organizations respond to failure & adapt to evolving conditions with grace



We can foster the five key ingredients we need
to brew the Resilience Potion during dev



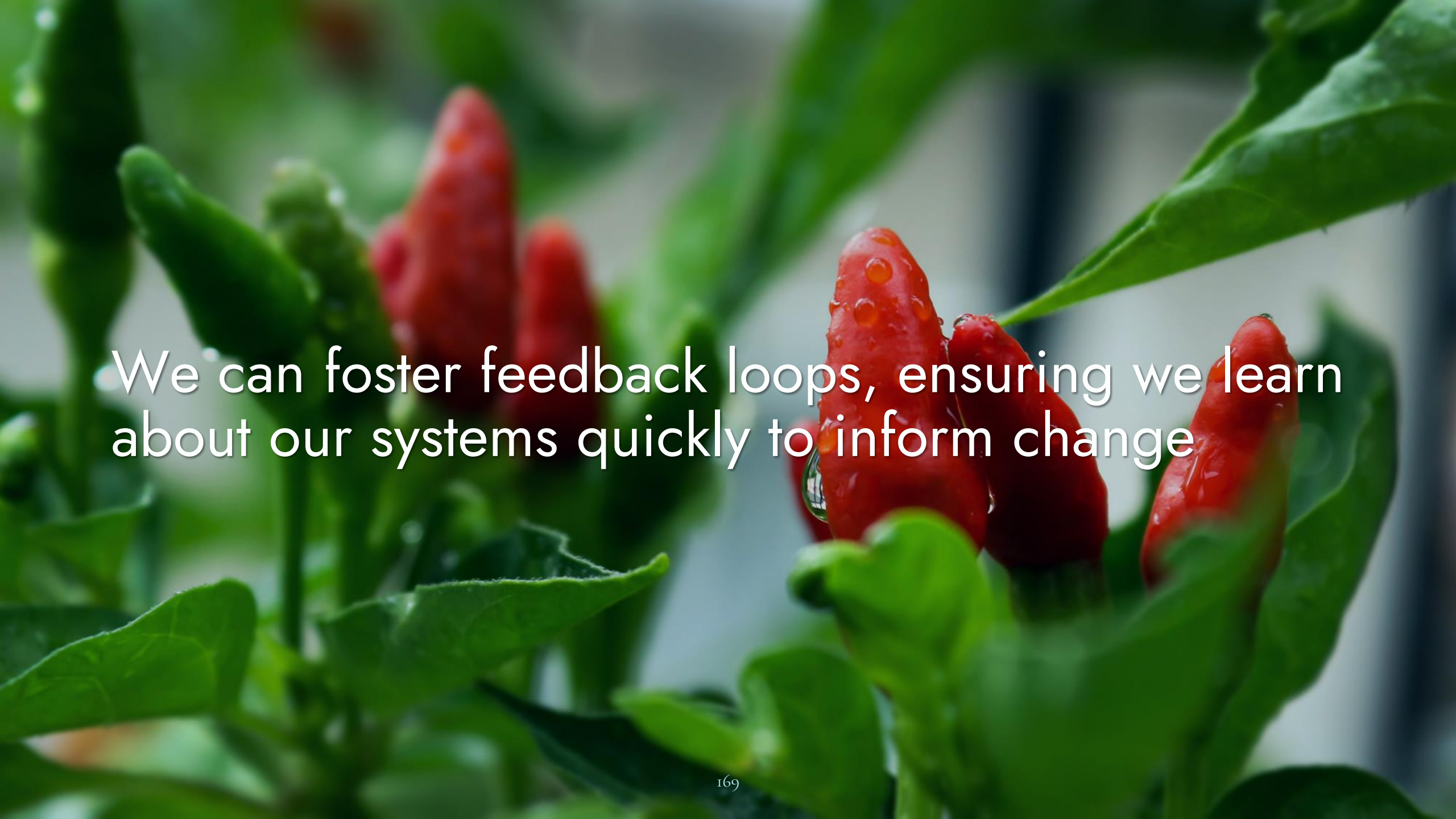
We can define our critical functions and
prioritize preserving them in adverse conditions



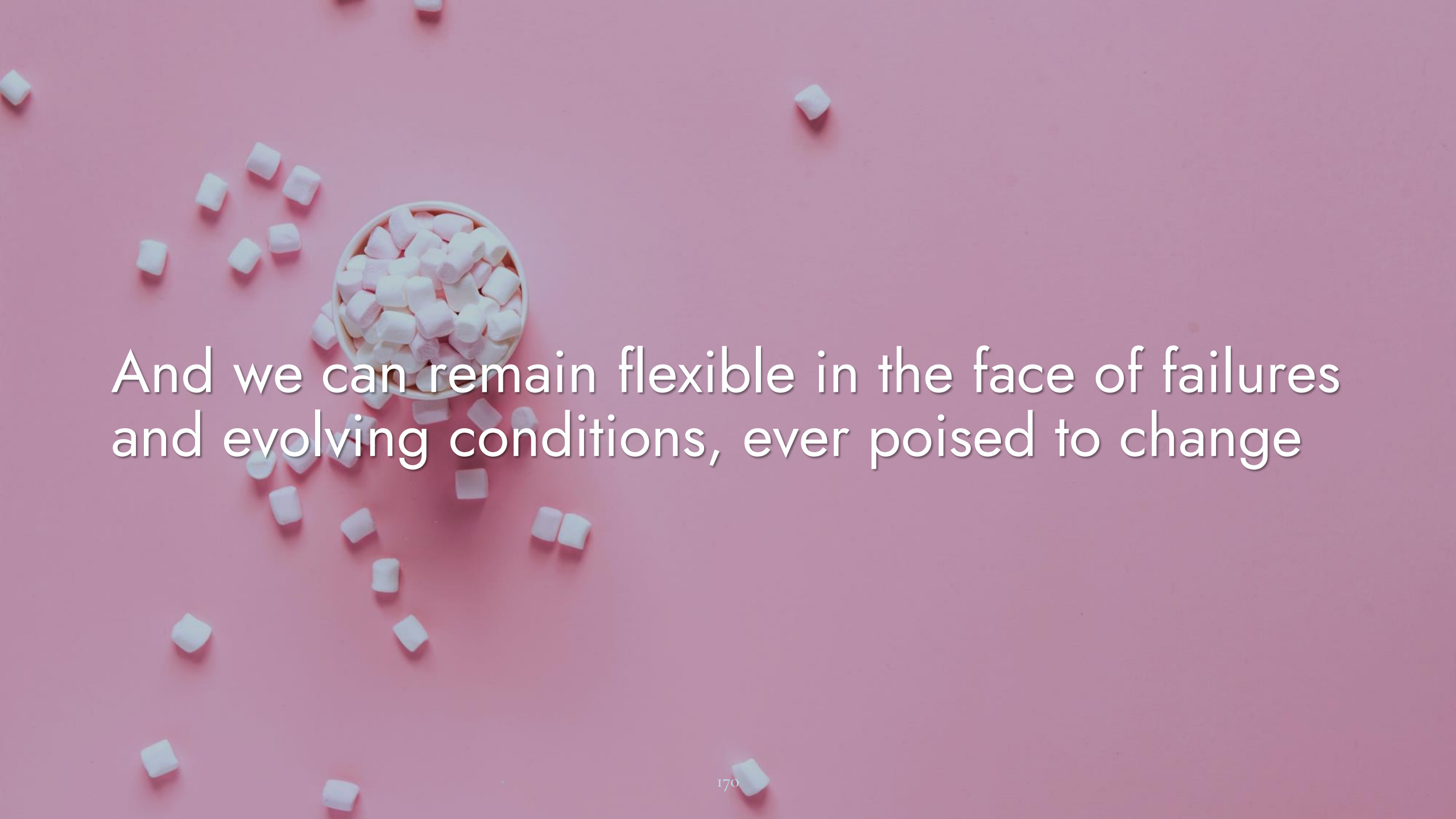
We can understand and expand our system's
boundaries of safe operation

A close-up photograph of almonds, almond skins, and a chocolate bar. The image is divided into three main sections: the left side is filled with whole almonds of various shades of brown; the top right is a dense pile of dark brown almond skins; and the bottom right features a portion of a dark chocolate bar with its characteristic square segments. The text "We can observe system interactions across space-time and make them more understandable" is overlaid in white, centered across the middle of the image.

We can observe system interactions across
space-time and make them more understandable

A close-up photograph of several bright red chili peppers on a green plant. The peppers are covered in small, clear water droplets, suggesting they have been recently watered or are in a humid environment. The background is softly blurred, showing more green foliage.

We can foster feedback loops, ensuring we learn about our systems quickly to inform change



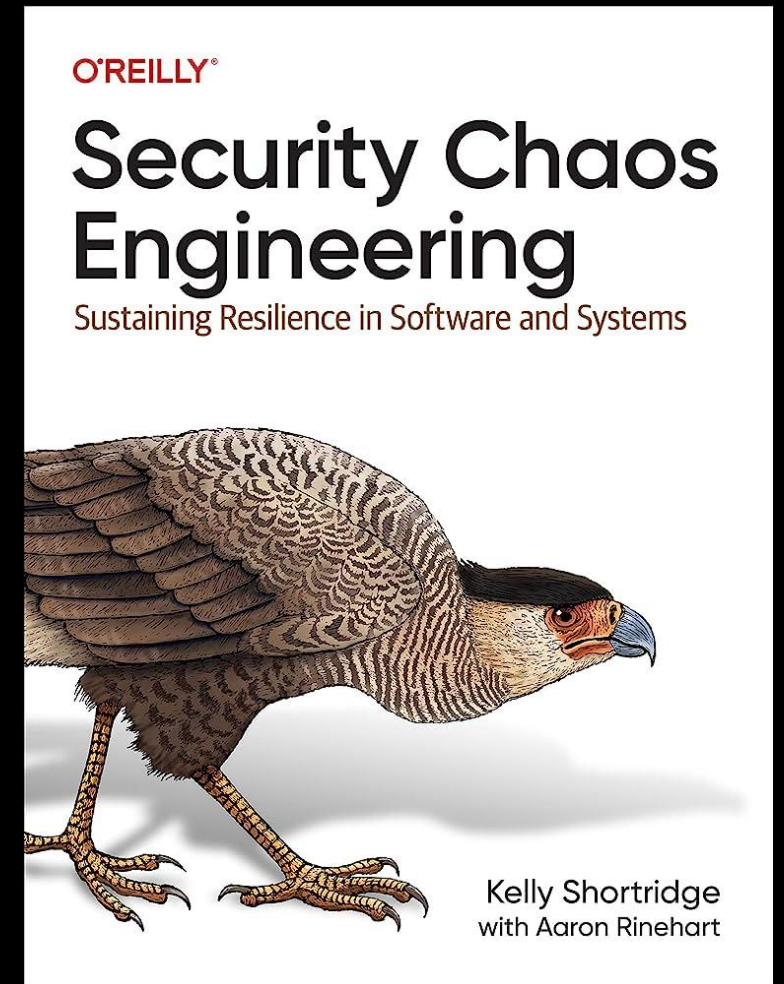
And we can remain flexible in the face of failures
and evolving conditions, ever poised to change

Order the book today:

[Amazon](#)

[Bookshop](#)

& other major retailers





/in/kellyshortridge



@swagitda_



shortridge@hachyderm.io



@shortridge.bsky.social



chat@shortridge.io