encryption and creating digital signatures. The first data may identify a particular one of the cryptographic key sets used by the first hardware component to decrypt the first data, and wherein the associated second metadata tag of the second data may identify a specific one of the cryptographic key sets used by the second hardware component to encrypt and sign the second data.

In accordance with another aspect of techniques herein is a method of processing a current instruction comprising: receiving, for metadata processing, the current instruction; and performing metadata processing for the current instruction in a metadata processing domain isolated from a code execution domain including the current instruction, said current instruction referencing a first memory location having a first metadata tag used in the metadata processing, said metadata processing for the current instruction including: performing processing to retrieve the first metadata tag from memory; prior to receiving the first metadata tag for the first memory location from the memory, determining a predicted value of the first metadata tag of the first memory location; determining, using the predicted value of the first metadata tag of the first memory location, whether the current instruction is allowed; and receiving, from the memory, the first metadata tag; determining whether the first metadata tag matches the predicted value of the first metadata tag; and responsive to determining the first metadata tag matches the predicted value of the first metadata tag, using the first result metadata tag as a final result metadata tag for the result operand. The metadata processing for the current instruction may include determining, in accordance with the current instruction and a set of input metadata tags for the current instruction, a first rule for the current instruction, wherein said first rule includes the predicted value of the first metadata tag of the first memory location and includes the first result metadata tag, said first rule being included in a rule cache used for metadata processing in the metadata processing domain; and responsive to determining the first metadata tag does not match the predicted value of the first metadata tag, performing rule cache miss processing in the metadata processing domain

computer readable medium comprising code stored thereon that, when executed, performs a method of processing a current instruction comprising: receiving, for metadata processing, the current instruction; and performing metadata processing for the current instruction in a metadata processing domain isolated from a code execution domain including the current instruction, said current instruction referencing a first memory location having a first metadata tag used in the metadata processing, said metadata processing for the current instruction including: performing processing to retrieve the first metadata tag from memory; prior to receiving the first metadata tag for the first memory location from the memory, determining a predicted value of the first metadata tag of the first memory location; determining, using the predicted value of the first metadata tag of the first memory location, whether the current instruction is allowed; and receiving, from the memory, the first metadata tag; determining whether the first metadata tag matches the predicted value of the first metadata tag; and responsive to determining the first metadata tag matches the predicted value of the first metadata tag, using the first result metadata tag as a final result metadata tag for the result operand.

BRIEF DESCRIPTION OF THE DRAWINGS

Features and advantages of the techniques herein will become more apparent from the following detailed description of exemplary embodiments thereof taken in conjunction with the accompanying drawings in which:

Figure 1 is a schematic drawing showing an example of a PUMP cache integrated as a pipeline stage in a processor pipeline;

Figure 2 is a schematic drawing showing a PUMP Evaluation Framework;

Figure 3A is a graph showing performance results for a single runtime policy with simple implementation using the evaluation framework depicted in Figure 2;

What is Claimed is

1. A method of processing a current instruction comprising:

receiving, for metadata processing, the current instruction; and

performing metadata processing for the current instruction in a metadata processing domain isolated from a code execution domain including the current instruction, said current instruction referencing a first memory location having a first metadata tag used in the metadata processing, said metadata processing for the current instruction including:

performing processing to retrieve the first metadata tag from memory;

prior to receiving the first metadata tag for the first memory location from the memory, determining a predicted value of the first metadata tag of the first memory location;

determining, using the predicted value of the first metadata tag of the first memory location, whether the current instruction is allowed; and

receiving, from the memory, the first metadata tag;

determining whether the first metadata tag matches the predicted value of the first metadata tag; and

responsive to determining the first metadata tag matches the predicted value of the first metadata tag, using the first result metadata tag as a final result metadata tag for the result operand.

2. The method of Claim 1, wherein the metadata processing for the current instruction further comprises:

determining, in accordance with the current instruction and a set of input metadata tags for the current instruction, a first rule for the current instruction, wherein

said first rule includes the predicted value of the first metadata tag of the first memory location and includes the first result metadata tag, said first rule being included in a rule cache used for metadata processing in the metadata processing domain; and

responsive to determining the first metadata tag does not match the predicted value of the first metadata tag, performing rule cache miss processing in the metadata processing domain for the current instruction.

3. The method of Claim 2, wherein said rule cache miss processing in the metadata processing domain for the current instruction includes:

determining whether execution of the current instruction in the code execution domain is allowed;

responsive to determining execution of the current instruction in the code execution domain is allowed, generating a new rule for the current instruction, wherein said new rule is generated in accordance with the current instruction, the set of input metadata tags, and the first metadata tag; and

inserting the new rule into the rule cache used for metadata processing in the metadata processing domain.

4. The method of Claim 2, wherein the set of input metadata tags include a plurality of other metadata tags for the current instruction, said set of other metadata input tags including metadata tags for any of: a program counter, the current instruction, and an input operand of the current instruction.

5. The method of Claim 1, wherein said result operand is a destination memory location or a destination register storing a result of executing the current instruction.

6. The method of Claim 2, wherein the instruction is processed in accordance with a plurality of stages including a first stage and a second stage wherein the first stage occurs prior to the second stage, wherein the predicted value of the first metadata tag of

the first memory location is determined in the first stage, and the second stage includes performing said determining whether the first metadata tag matches the predicted value of the first metadata tag, and the second stage also includes performing said rule cache miss processing in the metadata processing domain for the current instruction responsive to determining the first metadata tag does not match the predicted value of the first metadata tag.

7. The method of Claim 2, wherein the rule cache is configurable to operate in either a prediction mode or a normal processing mode in accordance with a prediction mode selector, said rule cache being configured to operate in the prediction mode when performing said metadata processing for the current instruction.

8. The method of Claim 7, wherein, when the rule cache is configured to operate in said prediction mode, the rule cache generates first outputs in accordance with the first rule, the first outputs including a metadata tag for a program counter of a next instruction, the first result metadata tag for the result operand of the current instruction, and the predicted value of the first metadata tag as an output of the first stage, and wherein, when the rule cache is configured to operate in said normal processing mode, the rule cache generates second outputs in accordance with a second rule different from the first rule, wherein the second outputs do not include the predicted value of the first metadata tag, and the second outputs include metadata tags for result operand of the current instruction and for the program counter of the next instruction.

9. The method of Claim 7, wherein the rule cache uses a first version of rules of a first policy when operating in the prediction mode and otherwise uses a second version of rules of the first policy when operating in the normal processing mode, and wherein the first rule is included in the first version of rules and the second rule is included in the second version of rules.

10. The method of Claim 1, further comprising:

determining, using the predicted value of the first metadata tag of the first memory location, a first result tag for a result operand of the current instruction.

11. A system comprising:

a pipeline processor including a plurality of pipeline stages, said plurality of stages including a memory stage and a writeback stage;

a programmable unit for metadata processing (PUMP) integrated that operates prior to completion of the memory stage the memory stage, wherein the PUMP performs metadata processing for a current instruction referencing a first memory location having a first metadata tag used in the metadata processing, wherein the PUMP receives first inputs including first metadata tags for the current instruction and wherein the PUMP generates first outputs provided as inputs to the writeback stage, the first outputs including a predicted value of the first metadata tag of the first memory location and a first result metadata tag for a result operand of the current instruction, wherein the first result metadata tag is determined by the PUMP in accordance with the predicted value of the first metadata tag for the first memory location; and

hardware components of said writeback stage that determine whether the first metadata tag for the first memory location matches the predicted value of the first metadata tag, and that use the first result metadata tag as a final result metadata tag for the result operand when the first metadata tag matches the predicted value of the first metadata tag.

12. The system of Claim 11, wherein the PUMP is a first PUMP that operates simultaneously with the memory stage and further operates in a prediction mode and determines the predicted value of the first metadata tag of the first memory location, and wherein the system includes a second PUMP that operates in a normal, non-prediction mode and does not determine any predicted value for the first metadata tag of the first memory location, the second PUMP integrated as another stage between the memory stage and the writeback stage.

13. The system of Claim 12, wherein the first PUMP uses a first version of rules of a

first policy for use when operating in the prediction mode, the second PUMP uses a second version of rules of the first policy for use when operating in the normal, non-prediction mode, the first PUMP determines the first outputs in accordance with a first rule from the first version, the second PUMP determines second outputs in accordance with a second rule from the second version, said second outputs include a second result metadata tag for the first memory location and said second outputs are provided as inputs to the writeback stage, and wherein the hardware components of the writeback stage additionally use the second result metadata tag as the final result metadata tag for the result operand when the first metadata tag does not match the predicted value.

14. A non-transitory computer readable medium comprising code stored thereon that, when executed, performs a method of processing a current instruction comprising:

receiving, for metadata processing, the current instruction; and

performing metadata processing for the current instruction in a metadata processing domain isolated from a code execution domain including the current

instruction, said current instruction referencing a first memory location having a first metadata tag used in the metadata processing, said metadata processing for the current instruction including:

performing processing to retrieve the first metadata tag from memory;

prior to receiving the first metadata tag for the first memory location from the memory, determining a predicted value of the first metadata tag of the first memory location;

determining, using the predicted value of the first metadata tag of the first memory location, whether the current instruction is allowed; and

receiving, from the memory, the first metadata tag;

determining whether the first metadata tag matches the predicted value of the first metadata tag; and

responsive to determining the first metadata tag matches the predicted value of the first metadata tag, using the first result metadata tag as a final result metadata tag for the result operand.